Oracle® Database

Installation and Administration Guide

11*g* Release 2 (11.2) for Fujitsu BS2000/OSD

E27508-03

July 2014

ORACLE®

Oracle Database Installation and Administration Guide, 11*g* Release 2 (11.2) for Fujitsu BS2000/OSD

E27508-03

Copyright © 2007, 2014, Oracle and/or its affiliates. All rights reserved.

Primary Author: Bharathi Jayathirtha

Contributing Author: Janelle Simmons

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are "commercial computer software" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information on content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services.

Contents

Preface xi

Audience xi
Using Oracle Database Documentation xi
Documentation Accessibility xii
Related Documents xii
Conventions Used in this Manual xii

1 Release Notes

1.1 Oracle Database Editions for BS2000/OSD 1-1
1.2 Options Not Supported 1-1
1.3 Features not Supported 1-2
1.4 Known Restrictions, Problems, and Workarounds 1-2
1.4.1 General Notes 1-3
1.4.2 Oracle Spatial 1-3
1.4.3 Oracle Real Application Testing 1-3
1.4.4 ALTER DATABASE … RESIZE 1-3
1.4.5 IEEE Standard Floating Point Number Restrictions 1-3
1.4.6 SQL*Loader 1-3
1.4.7 Oracle Scheduler 1-4
1.4.8 INIT.ORA Parameters 1-4
1.4.9 Archiving to Tape 1-4
1.4.10 Import and Export 1-4
1.4.11 Oracle Data Pump Import and Oracle Data Pump Export 1-4
1.4.12 Globalization Support 1-4
1.4.13 Oracle Call Interface 1-4
1.4.14 SQL*Plus 1-4
1.4.15 ADRCI 1-5
1.4.16 Connecting from Oracle Database 10g clients 1-5
1.4.17 Utilities in the POSIX environment 1-5
1.4.18 Oracle Net Services 1-5
1.4.19 Oracle Net BEQ Protocol 1-5
1.4.20 Oracle Protocol Support for TCP/IP 1-5
1.4.21 POSIX Subsystem 1-6

2 Architecture and Implementation

2.1 Operating System Environment ... 2-1
2.1.1 File Systems ... 2-1
2.1.2 POSIX Shell ... 2-2
2.1.3 Processes ... 2-2
2.2 Basic Structures ... 2-2
2.2.1 Database Files and Log Files ... 2-2
2.2.2 Other Oracle Database Files ... 2-3
2.2.2.1 Initialization File ... 2-3
2.2.2.2 Server Parameter File ... 2-3
2.2.2.3 ORAENV File ... 2-3
2.2.2.4 Control Files ... 2-3
2.2.2.5 Message Files ... 2-3
2.2.3 Oracle-Managed Files ... 2-4
2.2.4 Bigfile Tablespaces ... 2-5
2.3 Two-Task Mode ... 2-5
2.4 Address Space Planning ... 2-5
2.4.1 Oracle Database Data Area Placement ... 2-5
2.5 Oracle Database Environment Definition File ... 2-6
2.5.1 Generating ORAENV ... 2-6
2.5.2 Oracle Environment Variables ... 2-7
2.5.3 Running ORAENV ... 2-7
2.5.4 POSIX Environment and ORAENV File ... 2-7
2.6 The ORALOAD Library ... 2-9
2.7 The ORAMESG library ... 2-9
2.8 User ID Requirements ... 2-9
2.8.1 Installation User ID (ORAUID) ... 2-9
2.8.1.1 Authorizations and File Access Rights ... 2-9
2.8.1.2 Default Name ... 2-10
2.8.2 DBA User ID ... 2-10
2.8.2.1 Authorizations and File Access Rights ... 2-10
2.8.2.2 Default Name ... 2-11
2.8.3 User IDs for Oracle users ... 2-11
2.8.3.1 Authorizations and File Access Rights ... 2-11

3 Oracle Database Installation and Deinstallation

3.1 Overview of Oracle Database Installation ... 3-1
3.2 Planning the Installation ... 3-1
3.3 Oracle Database Preinstallation Requirements ... 3-2
3.3.1 Checking Hardware Requirements ... 3-2
3.3.1.1 Memory Requirements ... 3-2
3.3.1.2 System Architecture ... 3-3
3.3.1.3 Disk Space Requirements ... 3-3
3.3.1.4 Display Requirements ... 3-4
3.3.2 Checking Software Requirements ... 3-4
3.3.2.1 Operating System and Communication System Requirements ... 3-5
3.3.2.2 POSIX Parameters ... 3-5

3.3.2.3 Package Requirements ... 3-6
3.3.2.4 Additional BS2000 Software Components ... 3-7
3.3.2.5 Compiler and CRTE Requirements for Oracle Database Applications ... 3-7
3.3.2.6 Additional Software Requirements ... 3-8
3.3.3 Checking Network Setup ... 3-8
3.3.3.1 Checking Sockets subsystem ... 3-8
3.3.3.2 Checking BCAM Timer ... 3-8
3.3.3.3 Checking LWRESD ... 3-8
3.3.3.4 Checking Loopback Address ... 3-9
3.3.3.5 Checking the Configuration Files in the POSIX File System ... 3-9
3.3.4 Creating Required Operating System Users and Groups ... 3-9
3.3.4.1 Creating the BS2000 Installation User ID ... 3-9
3.3.4.2 Creating the POSIX System Group ... 3-9
3.3.4.3 Initializing the POSIX User ... 3-10
3.3.4.4 Creating Users and Groups for Oracle Databases ... 3-10
3.3.5 Identify Required Directories in POSIX ... 3-11
3.3.5.1 Oracle Base Directory ... 3-11
3.3.5.2 Oracle Inventory Directory ... 3-12
3.3.5.3 Oracle Home Directory ... 3-13
3.3.6 Identifying or Creating Oracle Base Directory in POSIX ... 3-13
3.3.6.1 Identifying an Existing Oracle Base Directory in POSIX ... 3-13
3.3.6.2 Create or Expand a File System for the Oracle Base Directory ... 3-14
3.4 Installing the Oracle Database Software ... 3-15
3.5 Oracle Database Postinstallation Tasks ... 3-17
3.6 Installing Multiple Oracle Systems ... 3-18
3.7 Removing Oracle Database Software ... 3-18

4 Creating and Upgrading a Database

4.1 Creating a Database ... 4-1
4.1.1 Pre-Creation Activities ... 4-1
4.1.2 Creating a Database Automatically ... 4-2
4.1.3 Creating a Database Manually ... 4-4
4.1.3.1 Copying the DBA Procedures ... 4-4
4.1.3.2 Creating the Database ... 4-5
4.1.3.2.1 Pre-Allocating the Files ... 4-5
4.1.3.2.2 Modifying the Initialization File ... 4-6
4.1.3.2.3 Modifying the ORAENV File ... 4-6
4.1.3.2.4 Using SQL*Plus to Create the Database ... 4-6
4.1.3.3 Installing Data Dictionary Views ... 4-7
4.1.3.4 Installing Data Dictionary Views for PL/SQL ... 4-7
4.1.3.5 Installing Online Help Messages ... 4-7
4.1.3.6 Installing the Demo Tables ... 4-7
4.1.3.7 Installing the Sample Schemas ... 4-7
4.1.3.8 Verifying Successful Creation of the Database ... 4-7
4.1.3.9 Installing Oracle Text ... 4-8
4.1.3.10 Installing Java ... 4-8
4.2 Upgrading a Database ... 4-8

4.2.1 Upgrade from Version 9 ... 4-8
4.2.2 Upgrade from Version 10 ... 4-8
4.2.3 Postupgrade Activities ... 4-9

5 Administering Oracle Database

5.1 Using the SQL*Plus Utility ... 5-1
5.1.1 Starting the SQL*Plus Utility in BS2000 ... 5-1
5.1.2 Calling SQL*Plus from a Procedure ... 5-2
5.1.3 Using SQL*Plus in the POSIX shell ... 5-2
5.1.4 Running System Commands from SQL*Plus ... 5-2
5.1.5 Using an Editor in SQL*Plus ... 5-3
5.1.6 Customizing SQL*Plus Profiles ... 5-3
5.2 Startup and Parameter Files ... 5-3
5.2.1 The Environment Definition File ORAENV ... 5-4
5.2.2 The Initialization File INIT.ORA ... 5-4
5.2.3 The Server Parameter File SPFILE ... 5-4
5.2.4 Using the Correct Initialization File ... 5-4
5.3 Remote Startup of a Database Instance ... 5-5
5.4 Automatic Diagnostic Repository ... 5-6
5.4.1 Automatic Diagnostic Repository Directories and Files ... 5-6
5.4.2 ADR Command Interpreter ... 5-7
5.5 Checking the Integrity of the Physical Data Structure ... 5-8

6 Backing Up and Recovering a Database

6.1 Backing Up an Oracle Database ... 6-1
6.1.1 Using BS2000 Utilities to Back Up an Oracle Database ... 6-1
6.1.2 Online Backup ... 6-2
6.2 Recovering Databases ... 6-2
6.3 Recovery Manager ... 6-3

7 Tuning Oracle Database

7.1 BS2000/OSD-Specific Parameters ... 7-1
7.1.1 Task Priority ... 7-1

8 openUTM Product Set

8.1 Operation of Oracle Database Using openUTM Programs ... 8-1
8.2 Distributed openUTM Files ... 8-1
8.3 Installing ... 8-2
8.4 Developing an Oracle Database/openUTM Application ... 8-2
8.4.1 Defining an Open String ... 8-4
8.4.1.1 Required Fields ... 8-5
8.4.1.2 Optional Fields ... 8-5
8.4.1.3 Examples ... 8-6
8.4.2 Using Precompilers with openUTM ... 8-7
8.4.2.1 Using Pro*C with the Default Database ... 8-7
8.4.2.2 Using Pro*C with a Named Database ... 8-7

8.5 Troubleshooting ... 8-8
8.5.1 Trace Files ... 8-8
8.5.1.1 Trace File Examples ... 8-8
8.5.2 Debugging ... 8-9
8.5.3 In-Doubt or Pending Transactions ... 8-9
8.5.4 Oracle Database SYS Account Tables ... 8-9

9 Oracle Net Services

9.1 Introducing Oracle Net Services ... 9-1
9.1.1 IPC Protocol Support ... 9-1
9.1.1.1 Overview of IPC ... 9-1
9.1.1.2 Using the IPC Protocol ... 9-2
9.1.2 TCP/IP Protocol Support ... 9-2
9.1.2.1 Overview of TCP/IP ... 9-2
9.1.3 Using the TCP/IP Protocol ... 9-3
9.1.4 Bequeath Protocol ... 9-3
9.1.4.1 Overview of the Bequeath Protocol ... 9-3
9.2 Shared Server Architecture ... 9-4
9.3 Oracle Advanced Security ... 9-5
9.4 Configuring the Network ... 9-6
9.4.1 Using Easy Connect Naming Method ... 9-7
9.4.2 Using the Local Naming Method ... 9-7
9.4.3 Using the Directory Naming Method ... 9-7
9.4.4 Configuration on the Server ... 9-7
9.4.5 Configuration on the Client ... 9-9
9.4.6 Testing the Configuration on the Client ... 9-9
9.5 Troubleshooting Oracle Net Services ... 9-9

10 Oracle Text

10.1 Installing Oracle Text ... 10-1
10.2 Starting Oracle Text utilities ... 10-1
10.3 Restrictions of Oracle Text on BS2000/OSD ... 10-2

11 External Procedures

11.1 Loading External Procedures ... 11-1

12 Java in the Database

12.1 Installation of a Java Enabled Database ... 12-1
12.2 Database character sets and Java Encodings ... 12-2
12.3 Java Demonstration Files ... 12-2

13 XML

13.1 Installation of Xdk ... 13-1
13.2 Features and Restrictions ... 13-1

14 Oracle Management Agent

14.1 Preinstallation Issues 14-1
14.2 Running the Installation Script 14-2
14.3 Running the Agent 14-3
14.4 Restrictions 14-4
14.5 Troubleshooting 14-4

15 Oracle on SQ Server

15.1 Concepts 15-1
15.2 Definitions 15-2
15.3 Product Set 15-2
15.4 Hardware Requirements 15-2
15.5 Software Requirements 15-2
15.6 Address Space Limit 15-3
15.7 Installing Oracle Database Software 15-3
15.8 Database Creation 15-3
15.9 Upgrade and Migration 15-3
15.10 Administration 15-3
15.11 Oracle application programs 15-3
15.12 openUTM Product Set 15-4

A Oracle Error Messages for BS2000/OSD

B Oracle Environment Variables

B.1 ORAENV Rules B-1
B.2 Built-in Variables B-2
B.2.1 LOGNAME B-2
B.2.2 ORAUID B-2
B.2.3 PGM B-2
B.2.4 TERM B-2
B.2.5 TSN B-2
B.3 General Variables B-2
B.3.1 CLN_BASE B-3
B.3.2 CLN_MPID B-3
B.3.3 DEFAULT_CONNECTION B-3
B.3.4 EXP_CLIB_FILE_IO B-3
B.3.5 IMP_CLIB_FILE_IO B-4
B.3.6 NLS_LANG B-4
B.3.7 OPS_JID B-4
B.3.8 ORASID B-4
B.3.9 PRINTPAR B-4
B.3.10 SQLPATH B-5
B.3.11 SSSIDPWF B-5
B.4 DBA Startup Variables B-5
B.4.1 Address and Size Specification B-5
B.4.2 BGJPAR B-6

B.4.3 BGJPRC_UID / BGJPRC_SID ... B-6
B.4.4 BGJ_LOG_JOBSTART ... B-6
B.4.5 sid_BGJPAR ... B-6
B.4.6 sid_USER ... B-6
B.4.7 user_ACCOUNT/ user_PASSWORD ... B-7
B.4.8 COM_MPID ... B-7
B.4.9 COM_BASE ... B-7
B.4.10 JOBID ... B-7
B.4.11 KNL_BASE ... B-7
B.4.12 ORACLE_HOME ... B-8
B.4.13 PGA_BASE ... B-8
B.4.14 PGA_SIZE ... B-8
B.4.15 SF_PBLKSIZE ... B-8
B.4.16 SGA_BASE ... B-9
B.5 Oracle Net Services Variables ... B-9
B.5.1 BREAK_HANDLING ... B-9
B.5.2 TNS_ADMIN ... B-9
B.5.3 TNS_BEQ_TIMEOUT ... B-9
B.5.4 TNS_UPDATE_IPNODE ... B-10

C Initialization Parameters and the Parameter File

C.1 Example Parameter File ... C-1
C.2 Unsupported Parameters ... C-1
C.3 Additional Notes on Initialization Parameters ... C-1
C.3.1 BACKGROUND_DUMP_DEST ... C-1
C.3.2 USER_DUMP_DEST ... C-2
C.3.3 AUDIT_FILE_DEST ... C-2
C.3.4 DB_BLOCK_SIZE ... C-2
C.3.5 DB_FILE_MULTIBLOCK_READ_COUNT ... C-2
C.3.6 DB_FILES ... C-3
C.3.7 LOCK_SGA ... C-3
C.3.8 SGA_MAX_SIZE ... C-3
C.3.9 LOG_ARCHIVE_BUFFER_SIZE ... C-3
C.3.10 LOG_ARCHIVE_DEST ... C-3

D Troubleshooting

D.1 Problems Installing Oracle Database 11*g* ... D-1
D.1.1 Problems Creating a Database ... D-1
D.2 Problems Starting a Database ... D-1
D.2.1 Problems Starting a Database ... D-1
D.2.2 Problems with Tasks ... D-2
D.3 Problems Accessing the Database ... D-2
D.3.1 Problems with Database and Log files ... D-2
D.3.2 Oracle Database 11*g* Trace Files ... D-3
D.3.3 Oracle Database-Level Error Information ... D-3

E File Types and Names Used by Oracle

Preface

This manual and *Oracle Database User's Guide for Fujitsu BS2000/OSD*, form a system-specific set of manuals that provide installation, maintenance, and usage information about Oracle Database and related products. This manual provides information for those responsible for installing and administering Oracle Database for BS2000/OSD and describes:

- How Oracle Database operates under BS2000/OSD
- How to install or upgrade Oracle Database
- How to create an Oracle Database

Oracle Database User's Guide for Fujitsu BS2000/OSD provides information for BS2000/OSD end-users of Oracle products. Certain topics are applicable both to users and database administrators.

This section contains the following topics:

- Audience
- Using Oracle Database Documentation
- Documentation Accessibility
- Related Documents
- Conventions Used in this Manual

Audience

This manual is written for people who are responsible for installing and maintaining Oracle Database, and for assuring its smooth operation. These people are usually system programmers who may have Database Administrator (DBA) responsibilities.

The reader is assumed to have a fundamental knowledge of BS2000/OSD. No attempt is made to document features of BS2000/OSD, except as they affect or are affected by Oracle Database.

Using Oracle Database Documentation

Oracle Database products that run under BS2000/OSD are identical, in the way in which they are supported, to Oracle Database products that run under any other operating system. However, because of the diversity of operating systems, the use of applications may differ slightly between different operating systems. As a result of this, Oracle provides two types of documentation:

Type	Meaning/Usage
Generic	This is the primary Oracle Database documentation, which describes how the product works and how it is used. Use this type of documentation to learn about product functions and how to use any Oracle Database product or utility.
System Specific	This documentation provides the information required to use the product under a specific operating system. Use this type of documentation to determine whether there are any system-specific deviations from the generic documentation.

This manual is written for users of Oracle Database for BS2000/OSD, providing them with BS2000/OSD-specific information about using Oracle Database products. It does not describe how to use a product unless its use is different than that described in the generic documentation. System programmers and database administrators responsible for installing the Oracle Database, or administering the Oracle Database, or both, should read this manual as well as *Oracle Database User's Guide for Fujitsu BS2000/OSD*. There are places where the information in these manuals overlap and is presented differently depending on the target audience.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at
http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Related Documents

For more information, refer to the following resources:

- Fujitsu documentation at http://manuals.ts.fujitsu.com/index.php?id=1
- *Oracle Database User's Guide for Fujitsu BS2000/OSD*

The latest documentation is available on OTN:

http://www.oracle.com/technetwork/indexes/documentation/index.html

Conventions Used in this Manual

The following conventions are observed in this manual.

Typographic Conventions

The following text conventions are used in this manual:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.

Convention	Meaning
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

Command Syntax

Item	Syntax
Commands	This font identifies text which must be entered exactly as shown: `set echo off`
Variables	Variables appear in italics. Substitute an appropriate value, for example: *arg1*
Required Items	Required items are enclosed in braces { }. You must choose one of the alternatives. `DEFINE { macro1 | macro2 }`
Optional Items	Optional items are enclosed in square brackets []. [*options*] *formname* [*userid/password*]
Repetitive Items	An ellipsis, ... represents an arbitrary number of similar items. CHKVAL *fieldname value1 value2... valueN*

Punctuation

The following symbols should always be entered as they appear in the command format:

Name	Symbol
ampersand	&
backslash	\
colon	:
comma	,
double quotation mark	"
equal sign	=
hyphen	-
number sign	#
parentheses	()
period	.
semicolon	;
single quotation mark	'

1

Release Notes

This chapter describes the features of Oracle Database on Fujitsu BS2000/OSD. The following topics are covered in this chapter:

- Oracle Database Editions for BS2000/OSD
- Options Not Supported
- Features not Supported
- Known Restrictions, Problems, and Workarounds

1.1 Oracle Database Editions for BS2000/OSD

Oracle Database is delivered in two editions for the two BS2000/OSD hardware lines:

- Oracle Database 11*g* Release 2 for Fujitsu BS2000/OSD (S series) is the edition for the BS2000/OSD servers based on /390 architecture (S Server)
- Oracle Database 11*g* Release 2 for Fujitsu BS2000/OSD (SQ series) is the edition for the BS2000/OSD servers based on x86-64 architecture (SQ Server)

Depending on the type of hardware, you must choose between these two editions. Oracle does not provide installation software specifically for the BS2000/OSD SX server. Use the Oracle Database 11*g* Release 2 for Fujitsu BS2000/OSD (S Series) software to install on an SX server. All information specific to the SQ series server is located in Chapter 15, "Oracle on SQ Server". All other chapters refer to both editions (the S and SX) server series.

1.2 Options Not Supported

This release implements the features of Oracle Database 11*g* Release 2 Enterprise Edition. These features are listed and described in the generic Oracle Documentation and in *Oracle Database Licensing Information* and *Oracle Database New Features Guide*.

The following options are not supported on Oracle Database 11*g* Release 2 for Fujitsu BS2000/OSD:

- Oracle Data Mining
- Oracle Database Vault
- Oracle TimesTen Application-Tier Database Cache (formerly known as Oracle In-Memory Database Cache)
- Oracle Label Security
- Oracle On-Line Analytical Processing (OLAP)

- Oracle RAC One Node
- Oracle Real Application Clusters (Oracle RAC)

1.3 Features not Supported

The following features are not supported on Oracle Database 11*g* Release 2 for Fujitsu BS2000/OSD:

- Oracle Universal Installer
- Oracle Automatic Storage Management
- Oracle Multimedia
- Oracle Globalization Development Kit
- Oracle XML DB
- Oracle Secure Backup
- Oracle Heterogeneous Services
- OCCI
- Oracle Instant Client
- Oracle Messaging Gateway
- Native Compilation of Java and PL/SQL
- JIT in Oracle JavaVM
- Cross Platform Transportable Tablespace
- Oracle Java Server Pages
- Oracle Application Express
- Oracle JDBC OCI drivers

1.4 Known Restrictions, Problems, and Workarounds

This section provides information about known restrictions and problems. It contains workarounds where possible and suggestions for certain common usage problems in the following sections. In addition to this section, refer to Appendix D, "Troubleshooting". If you encounter a problem that is not reported here, then contact Oracle Support for further assistance.

- General Notes
- Oracle Spatial
- Oracle Real Application Testing
- ALTER DATABASE … RESIZE
- IEEE Standard Floating Point Number Restrictions
- SQL*Loader
- Oracle Scheduler
- INIT.ORA Parameters
- Archiving to Tape
- Import and Export

- Oracle Data Pump Import and Oracle Data Pump Export
- Globalization Support
- Oracle Call Interface
- ADRCI
- Connecting from Oracle Database 10g clients
- Utilities in the POSIX environment
- Oracle Net Services
- Oracle Net BEQ Protocol
- Oracle Protocol Support for TCP/IP
- POSIX Subsystem

1.4.1 General Notes

The German characters ä, ö, ü and ß, cannot be used in the names of tables, columns, fields, synonyms, and so on. This is because these characters are converted into bracket symbols (for example, {). The characters can, however, be stored as data.

1.4.2 Oracle Spatial

The `SDO_GEORASTER` feature is not supported.

1.4.3 Oracle Real Application Testing

Workload Replay is not supported in this release.

1.4.4 ALTER DATABASE ... RESIZE

The operation `ALTER DATABASE ... RESIZE`, to make a database file smaller, is not supported on BS2000. This operation has no effect on the corresponding BS2000 files.

However, database files can be altered to larger sizes either manually, by using `ALTER` or automatically, when a tablespace is defined with `AUTOEXTEND`.

1.4.5 IEEE Standard Floating Point Number Restrictions

The `BINARY FLOAT` and `BINARY DOUBLE` data types are not supported for customer-written database applications running on BS2000/OSD.

Attempts to store or fetch these types from an application program running on BS2000/OSD produces unpredictable results with both local and remote Oracle databases.

1.4.6 SQL*Loader

The binary data types `FLOAT` and `DOUBLE` of SQL*Loader are not supported on BS2000/OSD. Using these data types with SQL*Loader and External Tables produces unpredictable results.

Use the external, non-binary data type FLOAT EXTERNAL instead.

Multithreading functionality of direct path loads is not supported on BS2000.

1.4.7 Oracle Scheduler

External Jobs are not supported.

1.4.8 INIT.ORA Parameters

A few initialization parameters in the `INIT.ORA` file, described in the generic documentation are not supported by Oracle Database 11*g* Release 2 for BS2000/OSD. Refer to Appendix C, "Initialization Parameters and the Parameter File" for more information.

1.4.9 Archiving to Tape

Archiving to tape is not supported; archive log files must always be created as disk files. You may, however, use normal BS2000 backup procedures to back up the archive log files created by the archive process.

1.4.10 Import and Export

This section provides information about Import and Export:

- Avoid ASCII/EBCDIC conversions by the operating system, FTP, or PERCON. Import and Export utilities perform their own conversions; additional conversions render the files unusable.
- Import and export on tapes are not possible to or from more than one tape.

1.4.11 Oracle Data Pump Import and Oracle Data Pump Export

This section provides information about Oracle Data Pump Import and Oracle Data Pump Export:

- Avoid ASCII/EBCDIC conversions by the operating system, FTP, or PERCON. Oracle Data Pump Import and Oracle Data Pump Export utilities perform their own conversions; additional conversions render the files unusable.
- Tapes are not supported with Data Pump Export and Data Pump Import.

1.4.12 Globalization Support

User-defined character sets implemented by Customizing Locale Data (as described in *Oracle Database Globalization Support Guide*) are not supported in this release.

1.4.13 Oracle Call Interface

The following features are currently not supported:

- OCI shared mode functionality
- OCI publish-subscribe functions
- OCI Thread package

1.4.14 SQL*Plus

The following remarks relate to SQL*Plus:

- SQL*Plus truncates and displays a warning message if the input lines exceed 511 characters.

- If you use the SQL*Plus `-s` option, then it must be the first option entered at the prompt.
- The internal message buffer is limited to 76 characters, therefore, certain messages are truncated. This typically occurs if a message includes a second message. In such cases, you usually refer to the message number part of the second message. For further information, refer to the manual relevant to the particular message, either *Oracle Database Error Messages*, or Appendix A, "Oracle Error Messages for BS2000/OSD" of this guide.
- If `ECHO` is set to `ON`, `TAB` is set to `ON`, and you specify a spool file, then the listing of commands may be misaligned.
- To define the POSIX editor `edtu` as your preferred text editor with SQL*Plus in the POSIX environment, the POSIX correction level A43 or higher is required.

1.4.15 ADRCI

If your preferred POSIX text editor is `edtu`, then the POSIX correction level A43 or higher is required.

1.4.16 Connecting from Oracle Database 10g clients

Oracle Database 10*g* clients can connect to Oracle Database 11*g* Release 2 instances on Fujitsu BS2000/OSD, if the initialization parameter `SEC_CASE_SENSITIVE_LOGON` is set to `FALSE`. Otherwise, the connection fails with the following message:

```
ORA-01017: invalid username/password; logon denied
```

1.4.17 Utilities in the POSIX environment

For utilities in the POSIX shell, such as SQL*Plus, Import or Export, the maximum length of path names is 54 byte.This restriction applies to the names of SQL scripts or spool files.

1.4.18 Oracle Net Services

The following remarks relate to Oracle Net Services:

- When you specify a name for the listener in the `LISTENER.ORA` file, Oracle recommends that the name is less than 20 characters long. If you use a listener name with more than 20 characters, then you must specify a log directory (trace directory) and log file (trace file).
- The listener can be started only if the POSIX subsystem is running.

1.4.19 Oracle Net BEQ Protocol

The support of the handoff/direct handoff technique makes the BEQ protocol incompatible to the BEQ protocol of releases prior to Oracle Database 10*g* Release 2. For example, an 11.2 listener cannot start a server of an Oracle9*i* instance.

1.4.20 Oracle Protocol Support for TCP/IP

The Internet Assigned Numbers Authority (IANA) divided port numbers into three ranges:

- Well Known Ports from 0 through 1023
- Registered Ports from 1024 through 49151

- Dynamic, or Private Ports, or both, from 49152 through 65535

Fujitsu documentation for TCP/IP on BS2000 recommends setting the privileged port to 2050. However, using a registered Oracle port number may cause conflicts (see http://www.iana.org/assignments/port-numbers). For example, if you set the port number for the listener process to 1521, then any Oracle process that listens on such a registered port number may fail with the following error:

```
TNS-12545: Connect failed because target host or object does not exist
TNS-12560: TNS:protocol adapter error
TNS-00515: Connect failed because target host or object does not exist
BS2000 Error: 126: Can't assign requested address
BS2000 BCAM-RC: 40010020
```

The workaround is to use a non-privileged port, or to set the privileged port number to a value less than 1500, usually to 1024.

1.4.21 POSIX Subsystem

Oracle Database 11*g* Release 2 requires the POSIX subsystem and access to the POSIX file system. The following exceptions may occur:

- Ask the BS2000 administrator to increase the number of UFS devices (parameter `NOSTTY` in `SYSSSI.POSIX-BC.`*`version`*), if you get the following message:

  ```
  CCM0090: ALL UFS TERMINAL DEVICES ARE IN USE OR PERMISSION DENIED
  ```

- The termination of the POSIX subsystem also terminates the Oracle instance.

2

Architecture and Implementation

This chapter describes the Oracle Database system architecture for Fujitsu BS2000/OSD. The chapter includes the following topics:

- Operating System Environment
- Basic Structures
- Two-Task Mode
- Address Space Planning
- Oracle Database Environment Definition File
- The ORALOAD Library
- The ORAMESG library
- User ID Requirements

2.1 Operating System Environment

Oracle Database 11*g* Release 2 runs on the BS2000/OSD operating system.

Oracle Database uses not only the interfaces of the *native* BS2000, but also the POSIX interfaces. This section includes the following:

- File Systems
- POSIX Shell
- Processes

2.1.1 File Systems

Oracle Database 11*g* Release 2 operates on two file systems, namely, BS2000 Data Management System (DMS) and the POSIX file system.

The POSIX file system is hierarchically structured and consists of directories and files.

As in earlier releases, most of the database files for Oracle Database 11*g* reside in the BS2000 DMS. For example, data files, control files, online and archive redo log files are placed in the BS2000 DMS.

Automatic Diagnostic Repository (ADR), the new feature, requires a hierarchical file system to create a file-based repository for database diagnostic data. So, the directories and files of ADR reside in the POSIX file system.

Therefore, Oracle Database 11*g* Release 2 for BS2000/OSD requires the POSIX file system. While in Oracle Database 10*g* the use of the POSIX file system was optional, it is mandatory in Oracle Database 11*g* Release 2.

During the installation of the Oracle Database software executables, libraries and other files are installed both in the BS2000 DMS and in the POSIX file system.

You must provide access to the POSIX file system. Refer to Chapter 3, "Oracle Database Installation and Deinstallation" for information about access to the POSIX file system.

2.1.2 POSIX Shell

Starting with Oracle Database 11*g* Release 2, you can start a subset of Oracle Database utilities not only in the BS2000 environment by using SDF commands, but also in the POSIX shell by using shell commands.

The diagnostic data in ADR can be viewed with the command line utility `ADRCI`. Start this utility in the POSIX shell.

Set the environment variable `ORACLE_HOME`, prior to starting utilities in the POSIX shell.

2.1.3 Processes

All processes of an Oracle Database instance, for example, dedicated server processes and background processes, run as BS2000 tasks.

If you start a client utility, such as SQL*Plus, in the BS2000 environment by using SDF commands, then the corresponding program is executed in the BS2000 login task.

If you start a client utility in the POSIX shell, then a new POSIX process is created.

Client processes running in the POSIX shell connect to an instance, such as clients in the BS2000 environment. The server process always starts in the BS2000 environment. For more information about processes in the BS2000 environment, refer to Chapter 9, "Oracle Net Services".

2.2 Basic Structures

The concepts of tasks (that is, processes in Oracle terminology) and memory structures (areas) are not BS2000 specific.

Refer to "Memory Architecture" in *Oracle Database Concepts* and "Process Architecture" in *Oracle Database Concepts* for detailed information.

This section includes the following:

- Database Files and Log Files
- Other Oracle Database Files
- Oracle-Managed Files
- Bigfile Tablespaces

2.2.1 Database Files and Log Files

One or more database files contain the data dictionary, the user data, and indexes.

Oracle Database requires a minimum of two log files that need not be the same size, although on BS2000/OSD, the recommended minimum is 10000 PAM blocks. The size of a log file is set in BS2000 blocks and not Oracle Database blocks.

Note: Both the BS2000/OSD operating system and Oracle Database perform input and output efficiently in units called blocks. A block is the basic unit of data storage. An Oracle Database block can be in one of the following formats:

- 2K, 4K, 6K, 8K, 16K, 32K when using BS2000 2K pubset format
- 4K, 8K, 16K, 32K when using BS2000 4K pubset format

Oracle Database and redo log files are BS2000 PAM files, and Oracle Database uses UPAM to access them.

2.2.2 Other Oracle Database Files

The following are the additional Oracle Database files:

- Initialization File
- Server Parameter File
- ORAENV File
- Control Files
- Message Files

2.2.2.1 Initialization File

The initialization file, `INIT.ORA`, contains a set of parameters that are read when an instance is started.

See Also: *Oracle Database Reference* and *Oracle Database Administrator's Guide* for more information about the initialization file

2.2.2.2 Server Parameter File

The server parameter file (`SPFILE`) is a binary server-side initialization file, which cannot be edited using a text editor. It is initially built from a traditional text initialization file using the `CREATE SPFILE` statement.

2.2.2.3 ORAENV File

Oracle Database utilities and products running in the native BS2000 environment use the Oracle Database environment definition file, which is referenced as `ORAENV`. This file contains the Oracle Database environment variables, which are used to describe the operating environment for each Oracle Database task. The database administrator also uses the `ORAENV` file to define BS2000-specific parameters necessary for database configuration.

2.2.2.4 Control Files

These files record the physical structure of a database and are specified in the initialization file.

2.2.2.5 Message Files

Message texts are stored in table modules, which are dynamically loaded from the `ORAMESG` library at execution time.

2.2.3 Oracle-Managed Files

The following is a list of the INIT.ORA parameters for oracle-managed files:

- DB_CREATE_FILE_DEST for data files, temp files, and block change tracking files
- DB_CREATE_ONLINE_LOG_DEST_n for redo log files and control files
- DB_RECOVERY_FILE_DEST for backups, archive log files, and flashback log files

On BS2000, these parameters are used as a prefix for file names.

Oracle tablespace names can be up to 30 characters long. To associate an OMF-created file name with its tablespace, you must use tablespace names that are distinct in the first eight characters. Oracle allows underscores(_) in tablespace names, and any underscores(_) that are present are changed to hyphens(-) for use in the generated file name.

File names for Oracle-managed files have the following format on BS2000:

File type	Format
control files	destOMC.tttttttt
log files	destOMLlll.tttttttt
permanent tablespace files, data file copies	destOMD.tsn.tttttttt
temporary tablespace files	destOMT.tsn.tttttttt
archive log files	destOMA.Tnnn.Snnnnnn.tttttttt
data file or archivelog backup piece	destOMB.Lnnn.tttttttt
rman autobackup piece	destOMX.xnnnnnnn.tttttttt
block change tracking files	destOMR.tttttttt
flashback log files	destOMF.tttttttt

where:

- *dest* is the destination string (_DEST) in the OMF parameter
- *tttttttt* is the encoded timestamp (which looks like a random mix of letters and numerals)
- *lll* is a three-digit log-group number
- *tsn* is up to eight characters for the tablespace name
- *Tnnn* is the letter "T" followed by a three-digit thread number
- *Snnnnnn* is the letter "S" followed by a six-digit sequence number
- *Lnnn* is the letter "L" followed by a three-digit incremental level
- *x* is the letter *P*, if the database has an SPFILE, or the letter *T* if the database does not have an SPFILE
- *nnnnnnn* is a seven-byte timestamp

Given the 54 character limit on BS2000 file names, the preceding file name formats impose a limit of 39 characters on DB_CREATE_ONLINE_LOG_DEST_n and DB_CREATE_FILE_DEST, 29 characters on DB_RECOVERY_FILE_DEST. This file name character limit includes the catid and userid, that may occupy up to 16 characters.

See Also: Refer to "Using Oracle Managed Files" in *Oracle Database Administrator's Guide* for more information about file name formats

2.2.4 Bigfile Tablespaces

Oracle Database 11*g* Release 2 on BS2000/OSD supports bigfile tablespaces. The single data file of a bigfile tablespace must reside on a BS2000 pubset with the following attributes

```
LARGE_VOLUMES=*ALLOWED and LARGE_FILES=*ALLOWED
```

See Also: Refer to "Files and Volumes Larger than 32 GB" for more information about handling large objects on BS2000/OSD at

http://manuals.ts.fujitsu.com/file/9118/dv32.pdf

2.3 Two-Task Mode

In two-task mode, a user task connects to a server task, which runs Oracle Database code on behalf of the user task. The user task does not have access to the SGA. Communication between a user task and a server task is through Oracle Net Services.

2.4 Address Space Planning

Oracle Database uses a number of data and code areas, which must be at the same virtual addresses in all server and background tasks. Typically, the default values provided with Oracle Database are sufficient. Address space planning (explicit placement of Oracle Database data areas) may be required in some special situations, when you encounter address space conflicts. For example, dynamic subsystems may occupy the default address ranges, which may require you to relocate Oracle Database areas.

2.4.1 Oracle Database Data Area Placement

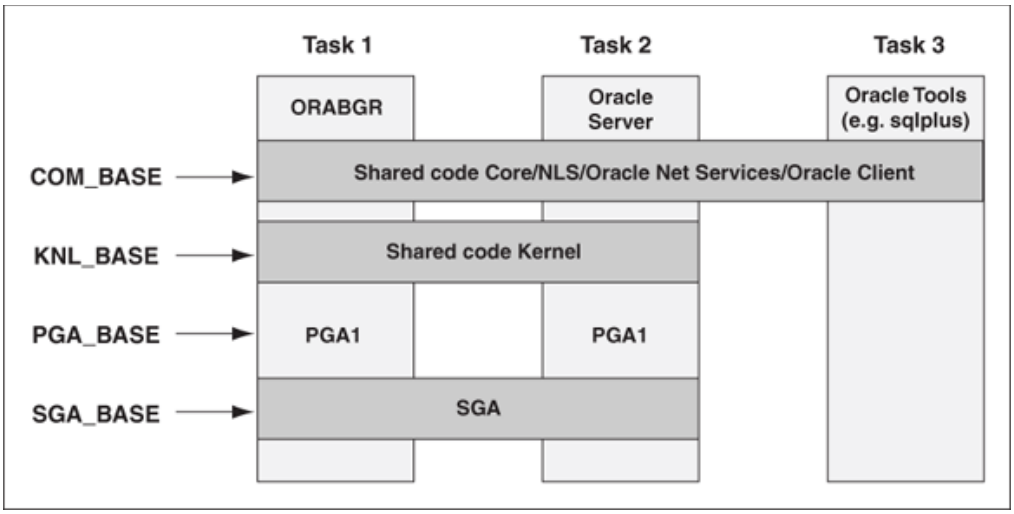
The following ORAENV variables control explicit placement of Oracle Database data areas:

- COM_BASE
- KNL_BASE
- PGA_BASE
- SGA_BASE

The order of the areas in the address space is not significant. The *xxx*_BASE variable is evaluated only during STARTUP processing.

After the database is started, users attaching to it do not need to specify the values in the ORAENV files, as they are automatically supplied with the common values during connection. This means that the settings in the user's ORAENV file are ignored. Figure 2–1 gives an example of the placement of data areas.

Figure 2–1 Placement of Data Areas in Background, Server and User Tasks



The *xxx*_BASE values must be compatible with the BS2000/OSD value SYSBASE (defined by BS2000/OSD generation and delimiting the user's address space).

Starting with Oracle Database 10*g*, user programs use a separate shared code pool for common services such as Core, Globalization Support, and Net Services. The name of this pool is `Client Common Pool` and its placement can be controlled by the `ORAENV` parameter `CLN_BASE`.

In general, Oracle administrators should be aware of conflicts between Oracle pool placements and other pool placements in the system.

2.5 Oracle Database Environment Definition File

This section describes the `ORAENV` file, how it is used, and how you use the environment variables to specify the default database.

The `ORAENV` text file has the format of a BS2000 command procedure that runs the `/SET-FILE-LINK ORAENV` command for itself. Each line contains an Oracle Database environment variable and its assigned value. When reading this file, the Oracle Database ignores all lines, which have a slash (/) or an asterisk (*) in column 1.

This section describes the following:

- Generating ORAENV
- Oracle Environment Variables
- Running ORAENV
- POSIX Environment and ORAENV File

2.5.1 Generating ORAENV

The `INSTALL.P.DBA` procedure automatically creates a copy of the `ORAENV` file. This file provides a default configuration for an Oracle Database. You can edit this file to adapt it to local needs. Users can also generate an `ORAENV` file specific to their own environment. This is described in the chapter "Getting Started" in *Oracle Database User's Guide for Fujitsu BS2000/OSD*.

2.5.2 Oracle Environment Variables

Appendix B, "Oracle Environment Variables" contains a list of Oracle environment variables that the database administrator can use. Most users only need to set a few of these variables. Any DBA-specific variables that are placed in a user's ORAENV file are ignored.

2.5.3 Running ORAENV

To set environment variables, simply run a CALL-PROCEDURE command on the ORAENV file containing the environment variables for the database you want to use. The name of the ORAENV file is *sid*.P.ORAENV (where *sid* is the database system identifier). For example, to set the environment variables for database DEMO using the example ORAENV file, run the following command:

```
/CALL-PROCEDURE DEMO.P.ORAENV
```

You can also generate an ORAENV file and run the /SET-FILE-LINK command before calling any Oracle Database program:

```
/SET-FILE-LINK ORAENV, filename
```

where *filename* is the name of a file having the same format as DEMO.P.ORAENV and which defines at least the ORASID environment variable.

Note:

- The database administrator should not modify the ORAENV file while the Oracle Database is running.
- Users may modify their ORAENV file at any time.

You can run several Oracle Databases simultaneously on your BS2000 system; even within the same Database Administrator account. A unique system identifier provides a globally unique name for the database so that a user can select a particular database by setting the ORASID environment variable. The user does this by activating the ORAENV file *sid*.P.ORAENV.

Whenever an Oracle Database product (for example, SQL*Plus) is started, it checks if the link name ORAENV is defined and reads the related file, storing the variable assignments for later use. If no link name ORAENV is set (or the related file cannot be read), then the *SID* remains undefined. Oracle recommends that a link name ORAENV is always defined prior to a call to an Oracle Database program.

2.5.4 POSIX Environment and ORAENV File

Every Oracle Database utility and product running in the POSIX shell get the environment variables from the POSIX environment. All Oracle and BS2000-specific variables can be set in the POSIX environment. The Oracle variable ORACLE_HOME, must be set. To run the utility for a particular database, you must also set the Oracle variable ORACLE_SID. The operating system environment variable PATH must be extended by the path to the Oracle binaries $ORACLE_HOME/bin. If you do not set any other Oracle variable in the POSIX environment, then the Oracle utilities use default values.The installation procedure creates a profile in the ORACLE_HOME directory which can be executed to set and expand the most important variables like ORACLE_HOME, PATH or LD_LIBRARY_PATH. You can process this profile with the command:

```
$ . your_oracle_home/.profile.oracle
```

Note: The Oracle variable BGJPAR is not set after running the .profile.oracle. If you do not set this variable, then the default value, BGJPAR=START=IMME,CPU-LIMIT=NO,LOGGING=*NO, is used by Oracle utilities.

You can change the default value of a variable by setting this variable in the POSIX environment. For example:

```
$ NLS_LANG=German_Germany.WE8BS2000
$ export NLS_LANG
```

Oracle also provides the opportunity to use the ORAENV file which you created in the BS2000 file system. In this case, you must create a file oraenv*sid* in the directory $ORACLE_HOME/dbs that describes the location and name of the ORAENV file.

Let us assume that you want to use the ORAENV file $ORADATA.ORCL.P.ORAENV. Create an oraenvORCL file in $ORACLE_HOME/dbs that contains the name of the BS2000 ORAENV file as follows:

```
$ ORACLE_HOME=/u01/app/orac1120/db11g
$ export ORACLE_HOME
$ echo '$ORADATA.ORCL.P.ORAENV' > $ORACLE_HOME/dbs/oraenvORCL
```

Set the Oracle environment variable ORACLE_SID and call the utility:

```
$ ORACLE_SID=ORCL
$ export ORACLE_SID
$ $ORACLE_HOME/bin/sqlplus /nolog
```

Oracle utilities running in the POSIX shell handle the variables of the BS2000 ORAENV file as lower-ranking variables. If a variable is set in the POSIX environment and the same variable is defined in the ORAENV file, then the POSIX variable is not overwritten by the ORAENV variable. For example, if the variable NLS_LANG is set to German_Germany.WE8BS2000 in the POSIX environment and NLS_LANG is also defined as American_America.WE8BS2000 in the BS2000 ORAENV file, then the variable keeps the value German_Germany.WE8BS2000 in the program environment of the Oracle utility running in the POSIX shell.

Note:

- The variables ORACLE_HOME and ORACLE_SID must be set.
- The ORAENV file in the BS2000 file system must be accessible for the utility.
- In a POSIX environment, the variables of the BS2000 ORAENV file are handled as subordinated variables.
- To access a BS2000 ORAENV file, you must create a file oraenv*sid* in the ORACLE_HOME/dbs directory that contains the fully qualified name of your BS2000 ORAENV file.
- The *SID* in the file name oraenv*sid* is case sensitive and must match the SID specified in ORACLE_SID.

2.6 The ORALOAD Library

The ORALOAD library ($ORAC1120.ORALOAD.LIB by default) is required to run any Oracle Database 11*g* Release 2 program. The Oracle Database uses this library to load executables and subroutines dynamically when required. The link name ORALOAD must identify the ORALOAD library before calling any Oracle Database program. If the link name is missing, then you get a BLS (BS2000/OSD loader) error message. Usually, this link name is set when the ORAENV procedure is called.

2.7 The ORAMESG library

The ORAMESG library ($ORAC1120.ORAMESG.LIB) is required for dynamically loading tables, such as message files, by an Oracle task when required. The link name ORAMESG must identify the ORAMESG library before calling any Oracle program. If the link name is missing, then you get a BLS (BS2000/OSD loader) error message. Usually, this link name is set when the ORAENV procedure is called.

2.8 User ID Requirements

Review the following sections to know about the user ID requirements.

- Installation User ID (ORAUID)
- DBA User ID
- User IDs for Oracle users

2.8.1 Installation User ID (ORAUID)

During installation, the complete Oracle Database software is installed into this user ID, which should be empty. This installation user ID (referred to as ORAUID) includes:

- executable programs, such as SQL*Plus, the background and network programs.
- load libraries, in particular, ORALOAD.LIB, from which modules are loaded during program execution. For example, the shared KERNEL module, and the precompiler run-time modules.
- message files.
- other data files, such as SQL file, for the DEMO tables.
- the INCLUDE files, application demo files, and system configuration files specifying default precompiler options for precompiler users.
- object libraries required to link-edit precompiler applications, such as PRO.LIB.
- port-specific installation utilities, such as programs, command procedures, and so on.
- default configuration files, such as the default ORAENV file.
- Oracle installation in the POSIX file system.

A separate ORAUID is required for each separate Oracle Database release. However, multiple databases using the same version can, and should, refer to the same installation user ID.

2.8.1.1 Authorizations and File Access Rights

This user ID does not require any special BS2000 privileges.

- You must not use the BS2000/OSD System Administrator user ID TSOS as an Oracle Database installation user ID.
- The ORAUID does not require any specific user attributes or privileges.
- Only the installation phase requires a BS2000 LOGIN under this user ID.
- During installation all files are created with the file attributes:

```
USER-ACC=ALL-USERS, ACCESS=READ
```

- You do not need to define write access for any file after installation.
- The installation in the POSIX file system requires a unique user number and group number for the installation user ID.

2.8.1.2 Default Name

The default name for the ORAUID is $ORAC1120. If you install Oracle Database 11*g* Release 2 under a different installation user ID, then the installation procedure changes the default value to the current installation user ID in all related files.

2.8.2 DBA User ID

The DBA user ID is a BS2000 user ID that is used as the owner of one or more Oracle databases. All the files for a specific Oracle database are owned by this user ID.

All tasks making up the running database, background tasks, and server tasks started for two-task Oracle Database, execute under the DBA user ID. These tasks refer to the executable programs and libraries, which are available under the installation user ID (ORAUID). These programs and libraries need not, and should not be copied into the DBA user ID. It is possible to use the installation user ID (ORAUID) as a DBA user ID. However, it is recommended that you use separate user IDs. The DBA user ID can also be used as a normal user ID.

Multiple databases can be created under the same, or under different DBA user IDs. If installed under different BS2000 user IDs, then the databases are separated and protected from each other, subject to the BS2000 protection mechanisms. In particular, a Database Administrator cannot administer a database running under a different BS2000 user ID (there is no global DBA privilege in Oracle Database for BS2000/OSD).

2.8.2.1 Authorizations and File Access Rights

The DBA user ID needs specific user attributes and privileges to run an Oracle Database. These privileges include:

- the right to start jobs immediately, preferably in a JOBCLASS reserved for Oracle Database background jobs. Failure to do this may cause delays when starting the database and when spawning server tasks.
- the right to start jobs with no time limit (TIME=NTL). Failure to do this may cause database tasks to terminate.
- the right to set jobs to TP state. Failure to do this may reduce database performance.
- the right to set Common Memory Pools as read-only. Failure to do this may reduce shared-code security.
- the BS2000/OSD System Administrator user ID TSOS should not, under any circumstances, be used as an Oracle Database DBA user ID.
- file access rights set under the DBA user ID should be:

```
USER-ACC=OWNER-ONLY, ACCESS=WRITE
```

- the POSIX user for the DBA user ID must be initialized with a unique user number and group number.

2.8.2.2 Default Name

There is no default name for the DBA user ID.

2.8.3 User IDs for Oracle users

An Oracle user accesses and uses the database through Oracle utilities, such as SQL*Plus, and through the precompiler application programs. The user can connect to Oracle Database through Oracle Net Services facilities.

The BS2000 user ID can also be used as Oracle Database connect user ID by the `OPS$` generic facility.

2.8.3.1 Authorizations and File Access Rights

These user IDs do not require any special BS2000 privileges.

- No file owned by a normal user needs any specific access attributes, as Oracle Database programs access such files locally from within that user ID. For example, `LOGIN.SQL` data files.
- No specific user attributes or privileges are needed.

3

Oracle Database Installation and Deinstallation

This chapter describes preinstallation requirements, postinstallation tasks, and the installation and deinstallation of Oracle Database 11*g* Release 2 on BS2000/OSD. It provides information about the following topics:

- Overview of Oracle Database Installation
- Planning the Installation
- Oracle Database Preinstallation Requirements
- Installing the Oracle Database Software
- Oracle Database Postinstallation Tasks
- Installing Multiple Oracle Systems
- Removing Oracle Database Software

3.1 Overview of Oracle Database Installation

The Oracle Database software can be downloaded from Oracle Software Delivery Cloud at `https://edelivery.oracle.com`.The installation files are included in a Library Maintenance System (LMS) library called staging library.

Download the zip file and extract it to a temporary location on a Windows or UNIX system. Then upload the staging library to the BS2000 system using File Transfer Protocol (FTP). The following sections describe the installation process for Oracle on the Fujitsu BS2000/OSD S Servers. The installation process for Oracle on the Fujitsu BS2000/OSD SQ Servers is identical, except for the name of the staging library. The following are the names of the staging library:

- `ora11202.s390.lib` for Fujitsu BS2000/OSD (S Series)
- `ora11202.sq.lib` for Fujitsu BS2000/OSD (SQ Series)

3.2 Planning the Installation

The Oracle Database installation process consists of the following steps:

1. **Read the release notes**: Read Oracle Database Release Notes for Fujitsu BS2000/OSD in Chapter 1, "Release Notes", before you begin the installation.
2. **Review the licensing information**: Although the installation media in the media pack contain many Oracle components, you are permitted to use only those components for which you have purchased licenses. Oracle Support Services does not provide support for components for which licenses have not been purchased.

> **See Also:** *Oracle Database Licensing Information* for more licensing information

3. **Complete preinstallation tasks**: "Oracle Database Preinstallation Requirements" describes preinstallation tasks that you must complete before installing the product.
4. **Install the software**: "Installing the Oracle Database Software" describes how to use the Oracle installation procedure for BS2000/OSD to install Oracle Database 11*g* Release 2.
5. **Complete postinstallation tasks**: "Oracle Database Postinstallation Tasks" describes recommended and required postinstallation tasks.
6. **Multiple software installations**: "Installing Multiple Oracle Systems" provides informations about multiple installations of Oracle software.
7. **Remove Oracle Database software**: "Removing Oracle Database Software" describes how to remove Oracle Database software from your system.

3.3 Oracle Database Preinstallation Requirements

This section describes the tasks that you must complete before you start the installation of Oracle Database 11*g* Release 2 for BS2000/OSD. It includes information about the following tasks:

- Checking Hardware Requirements
- Checking Software Requirements
- Checking Network Setup
- Creating Required Operating System Users and Groups
- Identify Required Directories in POSIX
- Identifying or Creating Oracle Base Directory in POSIX

3.3.1 Checking Hardware Requirements

The system must meet the following hardware requirements:

- Memory Requirements
- System Architecture
- Disk Space Requirements
- Display Requirements

3.3.1.1 Memory Requirements

Oracle Database requires at least 2 GB main memory.

The Address Space Limit for the installation user ID should be 512 MB or higher. For the DBA user IDs, it should be at least 1024 MB.

- To determine the memory size, enter the following command:

```
/SHOW-SYSTEM-INFORMATION INFORMATION=*MEMORY-SIZE
```

- To determine the user address space limit, log in to the desired user ID and enter the following command:

  ```
  /SHOW-USER-ATTRIBUTES
  ```

If the user address space is less than the required size, then ask your BS2000 system administrator to increase the Address Space Limit by using the command:

```
/MODIFY-USER-ATTRIBUTES USER-IDENTIFICATION=user-id,ADRESS-SPACE-LIMIT={512|1024}
```

3.3.1.2 System Architecture

Oracle Database 11*g* Release 2 supports all BS2000/OSD server architectures, such as S Server, SQ Server and SX Server. To determine the architecture on your system, enter the following command:

```
/SHOW-SYSTEM-INFORMATION INFORMATION=*HSI-ATTRIBUTES
```

The following is the TYPE output that shows the system architecture:

- IX: HSI of a S server (/390 architecture)
- KM: HSI of a SQ server (x86-64 architecture)
- PM: HSI of a SX server (SPARC architecture)
- UD: The HSI type is not defined

On S Servers, install the Oracle Database software for the platform Fujitsu BS2000/OSD (S Series). The name of the installation staging library is `ora11202.s390.lib`.

On SQ Servers, install the Oracle Database software for the platform Fujitsu BS2000/OSD (SQ Series). This software runs in native x86-64 mode. The name of the installation staging library is `ora11202.sq.lib`.

On SX Servers install the Oracle Database software for the platform Fujitsu BS2000/OSD (S Series). This software runs in emulated mode. The name of the installation staging library is `ora11202.s390.lib`.

3.3.1.3 Disk Space Requirements

Due to some new features, Oracle Database 11*g* Release 2 requires an installation in the BS2000 file system (DMS) and in the POSIX file system. Oracle recommends using a separate POSIX file system for the Oracle Database software installation.

> **See Also:** For more information about creating an additional POSIX file system, refer to the Fujitsu guide for BS2000/OSD mainframes, namely *POSIX Basics for Users and System Administrators* at:
>
> http://manuals.ts.fujitsu.com/file/8521/posix_g.pdf

When the POSIX administrator or BS2000 system administrator creates a new POSIX file system by using the `START-POSIX-INSTALLATION` command, a new POSIX container file similar to a UNIX disk partition is allocated in the BS2000 file system. The POSIX file system is created in this container file. If the desired mount point, for example, `/u01`, is not yet available in the root system, then it is created and the file system is mounted.

Oracle Database 11*g* Release 2 software for Fujitsu BS2000/OSD (S Series) requires the following disk space:

- BS2000 file system:

Item	Disk Space
Software	1.000.000 PAM pages
Staging library	820.000 PAM pages
Recommended (in total)	2.000.000 PAM pages

- POSIX file system:

Requirement	Disk Space
Minimum	400 MB
Recommended	1 GB

> **Note:** The diagnostic data of the Automatic Diagnostic Repository (ADR) reside in the POSIX file system. The disk space required for the Automatic Diagnostic Repository depends on the number of Oracle instances and the amount of diagnostic data created for incidents. So you may need significantly more disk space in the POSIX file system than recommended.

The extracted files occupy about 1.000.000 PAM pages in the BS2000 file system. The files installed in the POSIX file system occupy about 200 MB.

The staging library does not have to reside in your installation user ID.

3.3.1.4 Display Requirements

Oracle Database supports 9750 compatible terminals. In addition, Oracle Database supports X-terminals in the POSIX environment if you logged in to POSIX using `rlogin` or `ssh`.

The minimum resolution for Java based tools with a Graphical User Interface shipped with Oracle Database 11*g* Release 2 is 1024 x 768 or higher.

3.3.2 Checking Software Requirements

Before installing the Oracle Database software, it is recommended that you check if your system meets the following software requirements.

- Operating System and Communication System Requirements
- POSIX Parameters
- Package Requirements
- Additional BS2000 Software Components
- Compiler and CRTE Requirements for Oracle Database Applications
- Additional Software Requirements

3.3.2.1 Operating System and Communication System Requirements

The following operating system and communication system versions or higher, are required for Oracle Database 11*g* Release 2:

- BS2000/OSD V7.0
- *open*Net Server V3.3

  This includes BCAM V20.0 and Sockets V2.4.

Oracle recommends that you regularly install the most current correction packages for BS2000/OSD.

Check the version of your BS2000/OSD system with the following command:

```
/SHOW-SYSTEM-INFORMATION INFORMATION=*BS2000-ID
```

> **Note:** The correction level of the POSIX subsystem must be A41 or higher. You can check the POSIX correction level with the following command:
>
> ```
> /EXECUTE-POSIX-CMD CMD='uname -v'
> ```

You can check the version of your *open*Net Server software by checking the version of the software component BCAM with the following command:

```
/SHOW-INSTALLATION-PATH INSTALLATION-UNIT=BCAM
```

Or you can ask your BS2000 system administrator to check the BCAM version with the command:

```
/BCSHOW SHOW=BCAM
```

> **Note:** The BCAM version must be V20 or higher.

3.3.2.2 POSIX Parameters

The following POSIX parameters must be checked for the recommended values for Oracle Database 11*g* Release 2:

Parameter Name	Description	Recommended Value
HDSTNI	Number of hard disk server tasks	4
NPBUF	Number of physical I/O buffers	20
NPROC	Maximum number of processes	800
NBUF	Number of I/O buffers	200
MAXUP	Maximum number of processes per user	200
NOTTY	Maximum number of ttys	512
NOPTY	Maximum number of ptys	512

Parameter Name	Description	Recommended Value
NOSTTY	Maximum number of sttys	512
DBLSTATE	Initial state of POSIX loader	1
DBLPOOL	Size of pool in MB for POSIX loader	30

The POSIX administrator or BS2000 system administrator can check the values in the file `$TSOS.SYSSSI.POSIX-BC.`*`version`*, for example, version = 070 | 080 | 090. The values can also be checked by using the POSIX command `usp`:

```
# usp -s nproc
```

You can update the value with the following command:

```
# usp -P parameter -v value
```

Note:

- If you edit the parameters in the file `$TSOS.SYSSSI.POSIX-BC.`*`version`*, then you must restart the POSIX subsystem.
- The parameter values depend on the overall load of the BS2000 system and must be adjusted to this load. The values listed previously are minimal recommendations.

3.3.2.3 Package Requirements

The following POSIX packages with the specified versions or higher are required for Oracle Database 11*g* Release 2:

- POSIX-SH V7.0
- JENV V6.0
- APACHE V2.2, including correction package 2/2012
- PERL V5.8, including correction package 2/2012
- BCAM V20.0

To check the installed packages, use the following POSIX command:

```
$ pkginfo package name
```

Note:

- Ensure that the packages POSIX-SH and JENV are installed before starting the Oracle Database software installation.
- The packages APACHE, Perl and BCAM are required by the Oracle Management Agent. Although these packages are optional for Oracle Database 11*g* Release 2, it is recommended that you install these packages.

3.3.2.4 Additional BS2000 Software Components

The following additional software is required by Oracle Database 11*g* Release 2. Install the required or higher versions:

- CRTE-BASYS V1.7F or CRTE-BASYS V1.8E or CRTE-BASYS V1.9A
- SDF V4.7
- SDF-P V2.5
- SDF-A V4.1
- EDT V17.0A
- LMS V3.4
- openSM2 V7.0
- PTHREADS V1.3
- ORACLE-TOOLS V1.0

  The Oracle Database utility ADRCI requires the zip utility in the POSIX environment for creating ADR incident packages, that can be uploaded to Oracle Support.

  Currently, the zip utility is not part of the supply unit of POSIX of Fujitsu BS2000/OSD. So you must download it separately from the download area of the Fujitsu Support website:

  `http://support.ts.fujitsu.com/Download`

  Download the software package called ORACLE-TOOLS. This software package contains the utilities zip and unzip. You can find this software package by entering the package name `ORACLE-TOOLS` into the search box of the download area. Install the software package ORACLE-TOOLS as described in the readme file.

Note:

- The subsystem CRTEBASY must be loaded. On SQ server, the subsystem CRTEBASK must also be loaded.
- SDF-A is optional. SDF-A is only required to update the SDF user syntax files for Oracle. If SDF-A is not available, then the update of the SDF user syntax files for Oracle is skipped. If the SDF syntax files for Oracle are not updated, then you can start Oracle utilities with BS2000 SDF commands in UNIX-style, only if the installation user ID is the default installation user ID `ORAC1120`.
- The software components openSM2 and PTHREADS are required by the Oracle Management Agent.

3.3.2.5 Compiler and CRTE Requirements for Oracle Database Applications

If high-level languages, such as C or COBOL, are used to interface with the Oracle Database, then the following versions or higher are supported:

- COBOL85 V2.3
- COBOL2000 V1.2
- CPP V3.0
- CRTE V2.7F

- CRTE V2.8E
- CRTE V2.9

3.3.2.6 Additional Software Requirements

openUTM V6.0 or higher is required if you want to use Oracle Database in configurations with the transaction monitor openUTM.

3.3.3 Checking Network Setup

Typically, the computer on which you want to install Oracle Database is connected to the network as a member of a network domain. Besides the BCAM host name, the host should have a full qualified name (*hostname.domain*) that can be resolved by a DNS. This section describes the most important issues on a BS2000/OSD computer.

- Checking Sockets subsystem
- Checking BCAM Timer
- Checking LWRESD
- Checking Loopback Address
- Checking the Configuration Files in the POSIX File System

3.3.3.1 Checking Sockets subsystem

On S servers, the sockets subsystem SOC6 must be created and running. On SQ servers, the sockets subsystem SOC6-X8 must also be created and running. You can check the sockets subsystems with the following command:

```
/SHOW-SUBSYSTEM-STATUS SOC6*
```

3.3.3.2 Checking BCAM Timer

It is recommended to check the BCAM connection and letter timer. Your BS2000 system administrator can display the timer values with the following command:

```
/SHOW-BCAM-TIMER
```

The value for the connection timer, also called CONN in the output of the ISO section, should be set to a minimum of 600 seconds. It is recommended that the letter timer, also known as LETT in the STD section, should be set to a large value. The value `0` in the output indicates an infinite letter time.

3.3.3.3 Checking LWRESD

It is recommended to use the Light Weight Resolver (LWRESD) for host name resolution operations. The tasks for checking and administrating the LWRESD require the BS2000 system administrator privilege. Ask your BS2000 system administrator to check if the LWRESD is running by using the following command:

```
/SHOW-LWRESD-PARAMETERS
```

For example, the following output shows the parameter file in use:

```
RESOLV-FILE : :PVS1:$TSOS.SYSDAT.LWRESD.011.RESOLV.CONF
```

This parameter file must contain valid nameserver IP-addresses and a domain or search list of domains.

3.3.3.4 Checking Loopback Address

Check if the file `$TSOS.SYSDAT.BCAM.ETC.HOSTS` contains an entry for the loopback address. For example:

```
127.0.0.1               localhost loopback          # local address
```

3.3.3.5 Checking the Configuration Files in the POSIX File System

Check the network configuration files `/etc/hosts` and `/etc/resolv.conf`. These files must have the same contents as the corresponding files in the BS2000 administrator user ID TSOS. If there are any mismatches, then ask your BS2000 system administrator to synchronize the files.

3.3.4 Creating Required Operating System Users and Groups

If this is the first time Oracle software is installed on your system and depending on the installation of the Oracle Management Agent, you may have to create several BS2000 user IDs, POSIX users and groups. The BS2000 user ID for the Oracle Database 11*g* Release 2 software is called the installation user ID and the user ID where an Oracle database is stored is called the DBA user ID.

- Creating the BS2000 Installation User ID
- Creating the POSIX System Group
- Initializing the POSIX User
- Creating Users and Groups for Oracle Databases

3.3.4.1 Creating the BS2000 Installation User ID

The BS2000 System Administrator must create a user ID under which you want the Oracle Database 11*g* Release 2 software to reside. This user ID is called the Oracle installation user ID or Oracle software owner. Throughout this guide, we refer to this user ID as `ORAC1120`. This user ID does not require any special BS2000 privileges.

When the BS2000 system administrator creates the installation user ID with the following command, the corresponding POSIX user is also created. The POSIX user attributes, namely, user number, group number, login directory and program, are assigned default values.

```
/ADD-USER ORAC1120,…
```

3.3.4.2 Creating the POSIX System Group

For installation under POSIX, the POSIX administrator or BS2000 System Administrator must create system groups in POSIX. As distinguished from other UNIX systems, the installation user ID (Oracle software owner) and the DBA user ID of an Oracle database are not identical. Therefore, the POSIX installation uses only one group, such as `oracle`, for Oracle software owner and Oracle database. This group is called Oracle installation group.

- If this is not the first Oracle Database installation under POSIX, then you can determine the group name by using the following POSIX command:

  ```
  $ more /var/opt/oracle/oraInst.loc
  ```

  If the `oraInst.loc` file exists, then the output from this command is similar to the following:

  ```
  inventory_loc=/u01/app/oraInventory
  inst_group=oracle
  ```

The `inst_group` parameter shows the group name of your former Oracle Database installation, here `oracle`.

- To determine if this group is defined with a unique group-id, enter the following command:

```
$ grep 'oracle' /etc/group
```

If the output shows the group name and a group-id greater than 100, then the group exists.

If the group-id is 100, then you must change the id to a distinct value greater than 100, for example, 104.

If the group cannot be found in `/etc/group`, then ask your POSIX administrator or BS2000 system administrator to add the group name and a unique group number to the file `/etc/group`. The administrator must use a text editor, such as `edt` or `vi` to add a new line with the following specifications:

groupname::*groupnumber*:*user-id*[,*user-id*,...]

The following example shows a line in the file `/etc/group` for the group `oracle`:

```
oracle::104:ORAC1120,ORADB11G
```

Add further installation user IDs and all DBA user IDs to the line with the Oracle group-id.

3.3.4.3 Initializing the POSIX User

Before using the installation user ID as a POSIX user, the POSIX administrator or BS2000 system administrator must initialize the POSIX user with the following command:

/ADD-POSIX-USER USER-NAME=*user-id*,USER-NUMBER=*nnn*,GROUP-NUMBER=*oracle group number*,HOME-DIRECTORY=*path*[,RLOGIN-ACCOUNT=*account*]

Consider the following requirements:

- The `USER-NAME` is the installation user ID.
- The `USER-NUMBER` must be unique and greater than 100.
- The `GROUP-NUMBER` must be the number of the Oracle installation group and greater than 100.
- The `HOME-DIRECTORY` must be a valid path. Do not use a path which resides in the `root` or `var` file system. It is strictly recommended to create a separate file system for the POSIX users. An example for a valid `HOME-DIRECTORY` is `/home/orac1120` where `/home` is the mount point of a POSIX file system.
- The `RLOGIN-ACCOUNT` can be specified if you allow the remote access by using `rlogin` or `ssh`.

3.3.4.4 Creating Users and Groups for Oracle Databases

Typically Oracle databases in BS2000/OSD reside in BS2000 user IDs not equal to the Oracle database installation user ID. If you create a new DBA user ID for a new Oracle database, then you must also initialize the POSIX user in the same way as described in "Initializing the POSIX User". If you want to upgrade an existing Oracle database, then you must check if the POSIX user is initialized for the DBA user ID with the following BS2000 command:

```
/SHOW-POSIX-USER-ATTRIBUTES
```

If the output shows default values and the directory is `/home/gast`, then initialize the POSIX user with the following BS2000 command:

```
/ADD-POSIX-USER
```

The POSIX user attributes of an initialized POSIX user can be modified with the following BS2000 command:

```
/MODIFY-POSIX-USER-ATTRIBUTES
```

Finally, add the user to the Oracle installation group in the file `/etc/group`.

Note:

- Ensure that the home directory does not reside in the `root` or `var` file system.
- The `USER-NUMBER` must be unique and greater than 100.
- The `GROUP-NUMBER` must be the number of the Oracle installation group.

3.3.5 Identify Required Directories in POSIX

You must identify or create the following directories in the POSIX file system for the Oracle software:

- Oracle Base Directory
- Oracle Inventory Directory
- Oracle Home Directory

3.3.5.1 Oracle Base Directory

Oracle base directory is a top-level directory for Oracle software installations. Oracle recommends using a path similar to the following for the Oracle base directory:

```
/mount_point/app/software_owner
```

In this example:

- *`mount_point`* is the mount point directory for the file system that contains the Oracle software. The examples in this guide use `/u01` for the mount point directory.
- *`software_owner`* is the operating system user name of the software owner installing the Oracle software.

The components */mount_point/app* represent the operating system part of the path. Before you start the installation of Oracle Database 11*g* Release 2, you must create the operating system part of the path.

Note:

- The installation user ID and the installation group must have read, write and execute permissions to the subdirectory `app`.
- During Oracle Database software installation, you are prompted for a valid Oracle base directory. You can accept the default value or enter a new path. The installation procedure creates the Oracle base directory if possible.

You can use the same Oracle base directory for the installation of the Oracle Management Agent. Multiple Oracle installations require separate installation user IDs in BS2000/OSD with the consequence of creating separate Oracle base directories in the POSIX file system. The following are examples of Oracle base directories that can exist on the same system:

```
/u01/app/orac1120
/u01/app/ora10205
```

See "Create or Expand a File System for the Oracle Base Directory" for more information about the POSIX file system.

3.3.5.2 Oracle Inventory Directory

The Oracle Inventory directory (`oraInventory`) stores the inventory of all software installed on the system. It is required and shared by all Oracle software installations on a single system. If you have an existing Oracle Inventory path, then the Oracle installation procedure uses that Oracle Inventory.

The Oracle installation procedure derives the path for the Oracle Inventory directory from the file `/var/opt/oracle/oraInst.loc`. If this file does not exist, then it derives the inventory path from `ORACLE_BASE` and creates the Oracle Inventory directory in the following path: `ORACLE_BASE/../oraInventory`. For example, if `ORACLE_BASE` is set to `/u01/app/orac1120`, then the Oracle Inventory directory is created in the path `/u01/app/oraInventory`.

The Oracle installation procedure creates the Oracle subdirectories that you specify and sets the correct owner, group, and permissions for it. The operating system part of the path, for example, `/u01/app` must exist and you must have read, write and execute permissions in the directory `app`.

Note:

- All Oracle software installations rely on the Oracle Inventory directory. Ensure that you back it up regularly.
- Do not delete this directory unless you have completely removed all Oracle software from the system.
- By default, the Oracle Inventory directory is not installed under the Oracle base directory. This is because all Oracle software installations share a common Oracle Inventory, so there is only one Oracle Inventory for all software owners. Whereas, there is a separate Oracle base directory for each user.

3.3.5.3 Oracle Home Directory

The Oracle home directory is the directory where you install the software for a particular Oracle product. When you run the Oracle installation procedure, it prompts you to specify the path of this directory. You can accept the recommended path or enter a new path. The directory that you specify must be a subdirectory of the Oracle base directory. Oracle recommends that you specify a path similar to the following for the Oracle home directory:

```
oracle_base_path/product/dbhome
```

The Oracle installation procedure creates the directory path that you specify. It also sets the correct owner, group, and permissions on it. You do not have to create this directory manually.

> **Note:** During the installation, you must not specify an existing directory that has predefined permissions applied to it as the Oracle home directory. If you do, then you may experience installation failure due to file and group ownership permission errors.

3.3.6 Identifying or Creating Oracle Base Directory in POSIX

Before starting the installation, you must either identify the operating system part of an existing Oracle base directory or if required, create one. This section contains information about the following topics:

- Identifying an Existing Oracle Base Directory in POSIX
- Create or Expand a File System for the Oracle Base Directory

> **Note:** You can create an Oracle base directory, even if other Oracle base directories exist on the system.

3.3.6.1 Identifying an Existing Oracle Base Directory in POSIX

Existing Oracle base directories can be identified as follows:

- Identifying an existing Oracle Inventory directory.

  Search for the string `'inventory_loc'` in the file `/var/opt/oracle/oraInst.loc`.

  ```
  $ grep 'inventory_loc' /var/opt/oracle/oraInst.loc
  ```

  If `oraInst.loc` exists, then the output is similar to the following:

  ```
  inventory_loc=/u01/app/oraInventory
  ```

- Deriving an Oracle base directory from an existing Oracle home directory.

  Enter the following command to display the contents of the `/var/opt/oracle/oratab` file:

  ```
  $ more /var/opt/oratab
  ```

  If the `oratab` file exists, then it may contain lines similar to the following:

  ```
  *:/u01/app/orac1020/product/10g:N
  *:/oracl920/9.2.0:N
  ```

The directory paths specified on each line identify Oracle home directories. Directory paths that end with the user name of the Oracle software owner are existing Oracle base directories. For the installation of Oracle Database 11*g* Release 2, you can choose the subdirectory `/u01/app` as `root` for your Oracle base directory. If you intend to use the `orac1120` user to install the software, then you can choose the following directory:

```
/u01/app/orac1120
```

Ensure that the Oracle base directory for this installation meets the following conditions:

- It is not on the same file system as the operating system (`root` or `var`).
- It has sufficient free disk space.

To determine the free disk space on the file system where the Oracle base directory is located, enter the following command:

```
$ df -k oracle_base_path
```

Note: The installation procedure looks for an Oracle base directory and prompts you to accept the suggested path or to enter a new path.

3.3.6.2 Create or Expand a File System for the Oracle Base Directory

If you want to install the Oracle software into an existing file system with Oracle base directories but the POSIX file system has insufficient space, then the POSIX administrator or BS2000 system administrator can expand the file system with the Oracle base directory using the POSIX installer. Complete the following steps to expand the file system using the POSIX installer:

1. Start the POSIX Installer using the following command:

   ```
   /START-POSIX-INSTALLATION
   ```

2. Choose **Administrate POSIX filesystems**.
3. Mark the desired file system and choose **expand**.
4. Enter the number of PAM pages by which the file system should be expanded.

If you want the Oracle base directory to reside in a new POSIX files system, then the POSIX administrator or BS2000 system administrator must create a POSIX file system using the POSIX installer. Complete the following steps to create a POSIX file system using the POSIX installer:

1. Start the POSIX installer using the following command:

   ```
   /START-POSIX-INSTALLATION
   ```

2. Choose **Administrate POSIX filesystems** and then, choose **append**.

The POSIX installer allocates a new POSIX container file in the BS2000 file system and creates the POSIX file system inside the container. The POSIX installer also asks for a mount point for the new file system. If the mount point does not exist, then it is created in the root system and the new file system is mounted. By default, the owner of the new file system is SYSROOT. The administrator must now create the Oracle base directory and specify the correct owner, group, and permissions for it with the following specifications:

```
# mkdir -p /mount_point/app/oracle_sw_owner
```

```
# chown -R oracle_sw_owner:oracle_installation_group /mount_point/app/oracle_sw_
owner
# chmod -R 775 /mount_point/app/oracle_sw_owner
```

For example:

```
# mkdir -p /u01/app/orac1120
# chown -R orac1120:oracle /u01/app/orac1120
# chmod -R 775 /u01/app/orac1120
```

3.4 Installing the Oracle Database Software

The Oracle Database 11*g* Release 2 software is available in a zipped LMS library. Complete the following steps to install the Oracle Database software:

1. The software must be copied to a temporary location on a Windows or UNIX system that has FTP access to the BS2000 system where you are to install the software.

   Download the installation files from Oracle Software Delivery Cloud (https://edelivery.oracle.com/) and move the zip file to this Windows or UNIX system.

2. Unzip the zip file with a zip utility, such as WinZip, into a temporary location on this system. The following files should be created:

   `welcome.htm`: Starting point for accessing the documentation.

   `doc/`: Contains the platform specific documentation for Oracle Database on BS2000/OSD.

   And the installation files for one of the following platforms:

   – `ora11202.s390.lib`: Staging library for Fujitsu BS2000/OSD (S Series)

   – `ora11202.sq.lib`: Staging library for Fujitsu BS2000/OSD (SQ Series)

3. Transfer the staging library with FTP binary into a user ID on the BS2000 system. This can either be the Oracle installation user ID, for example, `ORAC1120`, or any other user ID. Before the transfer, if you are using BS2000 FTP, preallocate the file by using the following FTP file command:

   ```
   file ora11202.s390.lib,fcbtype=pam,blkctrl=no,blksize=(std,2),space=820000
   ```

   Use the following command if you are using FTP on the platform where you had unzipped the file:

   ```
   quote file
   ora11202.s390.lib,fcbtype=pam,blkctrl=no,blksize=(std,2),space=820000
   ```

4. If you have transferred the staging library to a user ID not equal to the Oracle installation user ID, then you must grant access to the staging library for all users, with the following command:

   ```
   /MODIFY-FILE-ATTRIBUTES ORA11202.S390.LIB,USER-ACCESS=*ALL-USERS
   ```

5. Log in to the installation user ID, for example, `ORAC1120`.

6. If the staging library resides in the installation user ID, then call the Oracle installation procedure as follows:

   ```
   /CALL-PROCEDURE (ORA11202.S390.LIB,ORAINST.PRC)
   ```

If you have stored the staging library in a different BS2000 user ID, then call the Oracle installation procedure as follows:

```
/CALL-PROCEDURE (staging_
userid.ORA11202.S390.LIB,ORAINST.PRC),(INSTUID=staging_userid)
```

For example:

```
/CALL-PROCEDURE ($FOO.ORA11202.S390.LIB,ORAINST.PRC),(INSTUID=$FOO)
```

The Oracle installation procedure performs the following actions:

- Checks hardware and software requirements
- Checks the available disk space for the BS2000 files and POSIX files
- Extracts the files from the staging library
- Installs Oracle software in the installation user ID
- Updates the SDF syntax files if SDF-A is available
- Installs Oracle Database software in the POSIX file system
- Registers the software in the Oracle Inventory file

The following is the complete syntax of the installation procedure:

```
/CALL-PROCEDURE ([staging_userid.]ORA11202.{S390|SQ}.LIB,ORAINST.PRC)[,
( [ CMD={INSTALL|UNINSTALL}]
[,INSTUID=staging_userid]
[,OLDUID=old_userid]
[,ORACLE-BASE={*DIALOG|oracle_base}]
[,ORACLE-HOME={*DIALOG|oracle_home}]
[,DEBUG={Y|N}]
[,PROCLOG={Y|N}] )]
```

Parameter	Value	Description
CMD	INSTALL|UNINSTALL default: INSTALL	action: install or uninstall Oracle Database software
INSTUID	user ID of staging library default: ''	BS2000 user ID where the staging library is located.
OLDUID	old user ID used in scripts default:$ORAC1120	User ID stored in several scripts when the software is released. Is updated to the current user ID during installation.
ORACLE-BASE	*DIALOG|oracle_base default: *DIALOG	The Oracle base directory is required for the installation of Oracle Database 11*g* Release 2 in the POSIX file system. The default value *DIALOG, indicates the prompting for an Oracle base directory.

Parameter	Value	Description
ORACLE-HOME	*DIALOG|oracle_home default: *DIALOG	The Oracle home directory is required for the installation of Oracle Database 11*g* Release 2 in the POSIX file system. The default value *DIALOG indicates the prompting for an Oracle home directory.
DEBUG	Y|N default: N	Debug the installation procedures.
PROCLOG	Y|N default: N	Use the logging feature of the BS2000 command CALL-PROCEDURE

The installation procedure also provides the opportunity to run the installation in a background task. In this case, you must specify the parameters ORACLE-BASE and ORACLE-HOME, otherwise the installation is aborted. Since the installation takes about 250 CPU seconds, Oracle recommends that you set the CPU-LIMIT to at least 300. You can start the Installation as follows:

```
/ENTER-PROCEDURE
(ORA11202.S390.LIB,ORAINST.PRC),(ORACLE-BASE='/u01/app/orac1120',ORACLE-HOME='/u01
/app/orac1120/product/dbhome'),CPU-LIMIT=300
```

The installation is logged to the file INSTALL.ORAINST.LST.

The installation of the Oracle Database software in the POSIX file system requires an ORACLE_BASE and ORACLE_HOME path. During installation you are prompted for these paths, if the installation runs in the dialog mode and if you have not specified the parameters in the procedure call. You can accept the suggested paths or enter new paths.

3.5 Oracle Database Postinstallation Tasks

To complete the installation, the POSIX administrator or BS2000 system administrator has to run the *oracle_home_path*/root.sh scripts as follows:

- In the BS2000 environment use the following command

  /EXECUTE-POSIX-CMD CMD='*oracle_home_path*/root.sh'

  For example,

  /EXECUTE-POSIX-CMD CMD='/u01/app/orac1120/product/dbhome/root.sh'

- In the POSIX environment (shell) use the following command

  # sh *oracle_home_path*/root.sh

  For example:

  # sh /u01/app/orac1120/product/dbhome/root.sh

If this is the first Oracle Database installation on the BS2000 server, then the root script completes the following actions:

- Creates the path /var/opt/oracle

- Creates the file `/var/opt/oracle/oraInst.loc`
- Creates an empty file `/var/opt/ortacle/oratab`
- If the product APACHE is installed, then it creates a link in the `lib` directory of the APACHE installation to the Oracle Client shared library in the Oracle home lib directory.

3.6 Installing Multiple Oracle Systems

You can also install multiple Oracle systems, based on the same or different versions of Oracle software. In this case, different versions of the software must be installed under different installation user IDs.

3.7 Removing Oracle Database Software

If you want to remove Oracle Database software from your system, then log in to the Oracle installation user ID and use the installation procedure as follows:

```
/CALL-PROCEDURE INSTALL.P.ORAINST,(CMD=UNINSTALL)
```

In this case, the Oracle installation procedure completes the following actions:

- Removes the Oracle Database software from the POSIX file system
- Updates the Oracle inventory
- Removes the Oracle Database software from the BS2000 file system

---

Note: Only the files installed by the Oracle installation procedure, namely, `ORAINST.PRC`, are removed from the system. Files created by a user or by an Oracle instance are left in the Oracle home directory and installation user ID.

4

Creating and Upgrading a Database

This chapter describes how to create a new 11.2 database or upgrade an existing database to release 11.2. It contains the following topics:

- Creating a Database
- Upgrading a Database

> **Note:** Creating a Java enabled database is not part of this chapter. For more information about creating a Java enabled database, refer to Chapter 12, "Java in the Database".

4.1 Creating a Database

You can create a database either automatically or manually. Oracle recommends that you use the automatic creation procedure outlined in "Creating a Database Automatically" on page 4-2. Instructions on how to create a database manually are given in "Creating a Database Manually" on page 4-4.

This section describes the following:

- Pre-Creation Activities
- Creating a Database Automatically
- Creating a Database Manually

4.1.1 Pre-Creation Activities

Before creating a database, you first need to carry out the following pre-creation activities:

1. Install Oracle Database 11*g* Release 2 software under the installation user ID. For details on how to do this, refer to Chapter 3, "Oracle Database Installation and Deinstallation".
2. The BS2000 System Administrator must create a `JOIN` entry for the account, which holds Oracle Database (the `DBA` user ID).

   The required privileges for this account are as follows:

   ```
   NTL=YES
   EXPRESS=YES
   TTYPL=TP
   CSTMP-MACRO=YES
   ```

Note: The value of ADDRSPACE must be at least 512 MB.

3. Oracle recommends that the BS2000 System Administrator should define a separate job class for the background tasks. This job class should have the following characteristics:

```
TP-ALLOWED=YES
NO-CPU-LIMIT=YES
JOB-TYPE=BATCH
```

4.1.2 Creating a Database Automatically

Complete the following steps to create a database automatically:

1. Log in using the DBA user ID.
2. To start the automatic creation procedure, INSTALL.P.SUPER, enter the following command:

```
/CALL-PROCEDURE $ORAC1120.INSTALL.P.SUPER
```

When running INSTALL.P.SUPER procedure, you can specify the value of the following keyword parameters (the default values are used if you choose not to modify the values):

Parameter	Value
BATCH	Enter **YES** to run the procedure in batch mode. The default is set to YES, so by default the procedure is run in batch mode.
CPULIMIT	Sets the time limit for batch jobs. The default is NO.
PLSQL	Enter **NO** to suppress automatic installation of the basic PL/SQL package. The default is YES.
VIEWS	Enter **NO** to suppress automatic installation of the basic views (catalog, import/export, and so on). The default is YES.

3. Answer the prompts for the following information (if you enter nothing, the default shown on the screen are used):

Parameter	Value
DBASID	Enter the 1 - 4 character system-id of the database you are installing. This is a mandatory parameter.
JOBCLASS	Enter the jobclass to be used for Oracle Database 11*g* Release 2 background jobs. This is mandatory.
UPDATE	Enter **YES** if you have existing files for this *SID* and you want to update them.
SYSPW	Enter the desired password for the Oracle Database user SYS. **Note**: By default the SYS user has the password CHANGE_ON_INSTALL. For security reasons, Oracle recommends that you change this password immediately after installation.
SYSTEMPW	Enter the desired password for the Oracle Database user SYSTEM. **Note**: By default the SYSTEM user has the password MANAGER. For security reasons, Oracle recommends that you change this password immediately after installation.

Parameter	Value
JAVA	Enter NO if you do not need a Java enabled database (thus saving memory, CPU and disk space resources). For more information refer to Chapter 12, "Java in the Database".
DBSIZE	Enter the size of the system tablespace files in bytes, kilobytes or megabytes. The value you enter can have one of the following forms: ■ 44M for 44 megabytes ■ 44000K for 44000 kilobytes ■ 10000000 for 10000000 bytes The default is 250M.
AUXSIZE	Enter the size of the sysaux tablespace file(s) in bytes, kilobytes, or megabytes. The value you enter can have one of the following forms: ■ 44M for 44megabytes ■ 44000K for 44000 kilobytes ■ 10000000 for 10000000 bytes The default is 200M.
LOGSIZE	Enter the size of the log files in bytes, kilobytes, or megabytes. The value you enter can have one of the following forms: ■ 1M for 1 megabytes ■ 1000K for 1000 kilobytes ■ 100000 for 100000 bytes The default is 20000K.
UNDOSIZE	Enter the size of the undo tablespace file in bytes, kilobytes, or megabytes. The value you enter can have one of the following forms: ■ 44M for 44 megabytes ■ 44000K for 44000 kilobytes ■ 10000000 for 10000000 bytes The default is 100M.
LOCAL	Enter **NO** if you do not require a locally managed system tablespace. The default is `YES`. If you choose a locally managed system tablespace, then Oracle automatically creates a default temporary tablespace.
DEFTS	Enter **NO** if you don't want to create a default permanent tablespace. The default is `YES`.
TEMPTS	This prompt only appears if you don't want a locally managed system tablespace. Enter **NO** if you don't want a default temporary tablespace. The default is `YES`.
CHARSET	Enter the character set with which you want the database to be created (the default is `WE8BS2000`). For more information refer to the chapter about Globalization Support in *Oracle Database User's Guide for Fujitsu BS2000/OSD*.
NCHARSET	Enter the national character set used to store data in columns specifically defined as `NCHAR`, `NCLOB`, or `NVARCHAR2`. Valid values are `AL16UTF16` and `UTF8`. The default is `AL16UTF16`.

Unless specified otherwise, `$ORAC1120.INSTALL.P.SUPER` generates and enter a batch job which:

- Calls `INSTALL.P.DBA`

- Creates the system and sysaux tablespace
- Creates the default permanent tablespace, temporary tablespace, and undo tablespace
- Creates the log files
- Initializes the database
- Runs `CATALOG.SQL`
- Runs `CATPROC.SQL`
- Installs the SQL Help tables
- Installs the DEMO tables
- Changes the system passwords if necessary
- Calls the verification procedure

When `$ORAC1120.INSTALL.P.SUPER` has completed, you should have an initialized, ready-to-use database, and a running Oracle Database system. The results of the job are listed in the file, `L.`*`sid`*`.INSSUP.SYSOUT`, where *`SID`* is the system ID of the database you have just installed.

4.1.3 Creating a Database Manually

Oracle recommends that you use the automatic creation procedure outlined in "Creating a Database Automatically" on page 4-2. The following manual creation procedure performs the same steps as the automatic creation procedure. However, because you enter the individual steps manually, you can perform the installation at your own pace, and determine which of the optional of the following steps you want to perform and which you want to expand, omit, or save for another time.

- Copying the DBA Procedures
- Creating the Database
- Installing Data Dictionary Views
- Installing Data Dictionary Views for PL/SQL
- Installing Online Help Messages
- Installing the Demo Tables
- Installing the Sample Schemas
- Verifying Successful Creation of the Database
- Installing Oracle Text
- Installing Java

4.1.3.1 Copying the DBA Procedures

Copy the DBA files from `$ORAC1120` as shown below:

1. Log in using the `DBA User ID`.
2. Call Oracle Database install procedure. This procedure copies the `DBA` files from `$ORAC1120` to the `DBA User ID` account. When the procedure begins you are prompted to supply a 1 to 4 character Oracle Database ID for the database you are installing.

   To install the `DBA` files, enter the following command:

```
/CALL-PROCEDURE $ORAC1120.INSTALL.P.DBA
```

This procedure prompts you for the following information:

Parameter	Value
DBASID	Enter the 1 - 4 character system ID of the database you are installing.
JOBCLASS	Enter the BS2000 jobclass to be used for background and server tasks.

You can also modify the following keyword parameters when invoking this procedure:

Parameter	Value
LOG	Enter **WRITE-TEXT** (the BS2000 command name) if you want to have install actions listed.
UPDATE	Enter **YES/NO** to indicate whether existing files are to be updated. The default is NO.

The `$ORAC1120.INSTALL.P.DBA` procedure copies the following files into the `DBA User ID` account:

- *sid*`.P.ORAENV`: Oracle Database environment definition file
- *sid*`.DBS.INIT.ORA`: Oracle Database initialization file

where *sid* is the database ID for the database being installed.

4.1.3.2 Creating the Database

After installing the DBA procedures, you must create the database. This section describes the procedure for creating the database, and for allocating the database file and the log files.

4.1.3.2.1 Pre-Allocating the Files You can place the database file and the log files on the default volume set for the `DBA` account, another Public Volume Set (PVS), or a specific private volume.

To create a database file or a log file on a private disk, you must first allocate the file using the BS2000 `/FILE` command as shown:

```
/FILE sid.DBS.DATABASE1.DBF,SPACE=filesize -
/[,DEVICE=device,VOLUME=volser]
/FILE sid.DBS.SYSAUX.DBF,SPACE=filesize -
//[,DEVICE=device,VOLUME=volser]
/FILE sid.DBS.LOG1.DBF,SPACE=filesize -
/[,DEVICE=device, VOLUME=volser]
/FILE sid.DBS.LOG2.DBF,SPACE=filesize -
/[,DEVICE=device, VOLUME=volser]
```

where:

sid identifies the database that you are installing.

filesize is the size of the file in PAM blocks. The file size specified in the `/FILE` command must match the size specified to SQL*Plus in the `CREATE DATABASE` statement when creating the database, plus 1 extra Oracle Database block used as an extra header. The size of this block is 1 to 16 PAM pages depending on the Oracle

Database block size given in the `init.ora` parameter `DB_BLOCK_SIZE`, refer to Appendix C, "Initialization Parameters and the Parameter File".

For example, if you want to create a 2 MB database file, then you must specify 1024 plus 1 PAM pages extra Oracle Database block as the value of file size in the `FILE` command.

`device` specifies the device to be used to store the file.

`volser` specifies the volume to be used to store the file.

The names used in the preceding examples are the default database and log file names. If you want to use other names, then remember to use these names in the SQL `CREATE DATABASE` statement, when creating the database.

4.1.3.2.2 Modifying the Initialization File Determine what changes, if any, you want to make to parameters in the distributed initialization file, *`sid`*`.DBS.INIT.ORA` (where *`sid`* is the database ID for the database). The SGA parameters may need to be adjusted to reflect memory limitations and the maximum number of users who can access the Oracle Database system at one time. Make the modifications using a BS2000 editor.

> **See Also:** Refer to *Oracle Database Reference* for an explanation of initialization parameters

4.1.3.2.3 Modifying the ORAENV File Modify the environment definition file, *`sid`*`.P.ORAENV`, according to the specific requirements. Remember that a number of variables are evaluated during startup only. If you modify the `ORAENV` file later on, then you may have to wait for the next startup for the changes to become effective.

The character set in the `ORAENV` variable `NLS_LANG`, however, must not be changed when you run some delivered SQL scripts.

4.1.3.2.4 Using SQL*Plus to Create the Database Remember that you must call the applicable *`sid`*`.P.ORAENV` procedure before calling SQL*Plus. To execute SQL*Plus, enter the following command:

```
/START-PROGRAM $ORAC1120.SQLPLUS
* /NOLOG
SQL> CONNECT / AS SYSDBA
SQL> STARTUP NOMOUNT [PFILE=filename]
```

`/NOLOG` omits being prompted for user name/password. `CONNECT` gives you a connection to an idle instance. The last statement starts the Oracle Database instance. If you want to use your own copy of the initialization file (*`sid`*`.DBS.INIT.ORA`), then use the `PFILE=`*`filename`* option, as illustrated in the previous command.

```
SQL> CREATE DATABASE...;
```

This statement creates database and log files.

Note: If you get an error before the first `SQL>` prompt, then it may be caused by either a missing `ORAENV` file (or `ORASID` not set in the `ORAENV`), or sometimes by an address space conflict. For example, the address range you assigned to the kernel memory pool (`KNL_BASE`) could be occupied by a subsystem.

4.1.3.3 Installing Data Dictionary Views

Data dictionary views provide easy access to dictionary information. If you want to use dictionary views, then you must install them by running SQL*Plus, by issuing the CONNECT / AS SYSDBA command, and entering the following commands:

```
SQL> SPOOL filename
SQL> SET TERMOUT OFF
SQL> @$ORAC1120.RDBMS.ADMIN.CATALOG.SQL
```

Data dictionary views required for Export or Import are also installed in this sequence.

4.1.3.4 Installing Data Dictionary Views for PL/SQL

If you chose not to install the basic PL/SQL package automatically while running INSTALL.P.SUPER, then you must perform the following task after the first STARTUP, in order to make PL/SQL available.

Run SQL*Plus, issue the CONNECT / AS SYSDBA command, and run the SQL script RDBMS.ADMIN.CATPROC.SQL to install the PL/SQL dictionary tables:

```
SQL>@$ORAC1120.RDBMS.ADMIN.CATPROC.SQL
```

4.1.3.5 Installing Online Help Messages

To install the online Help facility, enter the following command:

```
/CALL-PROCEDURE $ORAC1120.INSTALL.P.HELP,(sid [,SYSTEMPW=systempw])
```

4.1.3.6 Installing the Demo Tables

To install the demo tables, enter the following:

```
/CALL-PROCEDURE $ORAC1120.INSTALL.P.DEMO,(sid [,SYSTEMPW=systempw])
```

4.1.3.7 Installing the Sample Schemas

To install the Sample Schemas, enter the following:

```
/CALL-PROCEDURE $ORAC1120.INSTALL.P.SAMPLES,(sid -
/[,SYSTEMPW=systempw] [,SYSPW=syspw])
```

The procedure INSTALL.P.SAMPLES installs the sample schemas human resources (HR), order entry (OE), info exchange (IX) and sales history (SH) with the default passwords. product media (PM) is not supported.

> **Note:** By default, the password for the SYSTEM user is MANAGER and the password for the SYS user is CHANGE_ON_INSTALL. For security reasons, Oracle recommends that you change these passwords and the sample schema passwords immediately after installation.

4.1.3.8 Verifying Successful Creation of the Database

To verify that the demonstration database was correctly created, enter the following:

```
/CALL-PROCEDURE $ORAC1120.INSTALL.P.VERIFY,(sid [,SYSTEMPW=systempw])
```

If the demonstration database was correctly created, then you see messages like the following one displayed on the screen:

```
*SCOTT'S TABLE EMP IS INSTALLED
```

4.1.3.9 Installing Oracle Text

Usage and installation of Oracle Text is summarized in the "Oracle Text" chapter of this book.

4.1.3.10 Installing Java

Using and installing of Java is summarized in the chapter "Java in the Database" in this book.

4.2 Upgrading a Database

This section contains information about upgrading your Oracle database that are described in the following:

- Upgrade from Version 9
- Upgrade from Version 10
- Postupgrade Activities

4.2.1 Upgrade from Version 9

For upgrading from version 9 refer to "Upgrading to the New Release of Oracle Database" in *Oracle Database Upgrade Guide*.

4.2.2 Upgrade from Version 10

This section explains the BS2000 specific steps of the upgrade path. We assume the reader to be familiar with Oracle Database 11*g* Release 2 upgrade documentation about upgrade preparation, space and backup requirements, release differences, and so on. Refer to *Oracle Database Upgrade Guide* for further information regarding database upgrades. Oracle recommends reading the appropriate section in the generic documentation, especially when you are using `TIMESTAMP WITH TIME ZONE` data type.

We further assume your Oracle Database 10*g* is set up and the Oracle Database 11*g* Release 2 software is properly installed as explained in Chapter 3, "Oracle Database Installation and Deinstallation". Thereafter, complete the following steps:

1. Run the Pre-Upgrade Information Tool `utlu112i.sql` from Oracle Database 11*g* installation id with SQL*Plus in your Oracle Database 10*g* environment to analyze required parameters as follows (assuming the database is running):

```
/START-PROGRAM $ORACL1020.sqlplus
* /nolog
SQL> connect / as sysdba
SQL> SPOOL info.log
SQL> @$ORAC1120.rdbms.admin.utlu112i.sql;
SQL> SPOOL off
```

2. Check the sections in the spoolfile for Logfiles, Tablespaces, and Rollback Segments, and change the appropriate values of your database.
3. Shutdown the database and exit SQL*Plus

```
SQL> SHUTDOWN IMMEDIATE
SQL> exit
```

4. Enter the following command to create an Oracle Database 11*g* Release 2 `init.ora` and an Oracle Database 11*g* Release 2 `ORAENV` file and save the original files under the suffix `.OLD`.

```
/CALL-PROCEDURE $ORAC1120.INSTALL.P.DBA, (sid, jobclass, UPDATE=YES)
```

5. Modify the newly created files according to your special requirements (for instance `PROCESSES`, `DB_CACHE_SIZE`, and so on) and set the parameters in the `INIT.ORA` file as recommended in the spoolfile. Ensure that the `COMPATIBLE` initialization parameter is properly set for Oracle Database 10*g* Release 2 (10.2 or higher).

6. Set the Oracle environment variables using the `ORAENV` file:

```
/CALL-PROCEDURE sid.p.oraenv
```

   To avoid being prompted for many overflow acknowledgements on your screen, set the overflow to No:

```
/tchng oflow=no
```

7. Start the instance in upgrade mode:

```
/START-PROGRAM $ORAC1120.sqlplus
* /nolog
SQL> connect / as sysdba
SQL> STARTUP UPGRADE
```

8. Start the upgrade script:

```
SQL> SPOOL upgrade.log;
SQL> @$ORAC1120.rdbms.admin.catupgrd.sql;
SQL> SPOOL off
```

9. Shut down and restart the instance to re-initialize the system parameters for normal operation:

```
SQL> SHUTDOWN IMMEDIATE
SQL> STARTUP
```

10. Run the postupgrade script `utlu112s.sql` to display the results of the upgrade and run `catuppst.sql` to finish the upgrade. Finally run `utlrp.sql` to recompile any remaining stored PL/SQL and Java code and verify that all expected packages and classes are valid:

```
SQL> SPOOL postupgrade.log;
SQL> @$ORAC1120.rdbms.admin.UTLU112S.SQL;
SQL> SPOOL off
SQL> @$ORA11202.RDBMS.ADMIN.CATUPPST.SQL
SQL> @$ORAC1120.rdbms.admin.utlrp.sql;
SQL> SELECT count(*) FROM dba_objects WHERE status='INVALID';
SQL> SELECT distinct object_name FROM dba_objects WHERE status='INVALID';
```

    Now you should have an upgraded Oracle Database 11*g* Release 2 database. For troubleshooting, refer to "Upgrading to the New Oracle Database 10*g* Release" in *Oracle Database Upgrade Guide*.

4.2.3 Postupgrade Activities

This section describes the tasks you must complete after upgrading your database to Oracle Database 11*g* Release 2.

Re-compilation of C and COBOL Programmatic Interface Programs

All C and COBOL programs developed prior to release 11.2 must be precompiled using the new version of the precompilers and recompiled.

Re-Linking Programmatic Interface Programs

All user-written precompiler or Oracle Call Interface applications must be re-linked using the new Oracle Database libraries.

Rebuilding Oracle Database (openUTM) Applications

openUTM applications must be rebuilt. Refer to the chapters on openUTM in this manual and in *Oracle Database User's Guide for Fujitsu BS2000/OSD* for more information.

Updating ORAENV Files

Your `ORAUID` environment variable must reference the correct Oracle Database installation user ID. Check your `ORAENV` files, and if necessary, amend the values of the `ORAUID` and `NLS_LANG` environment variables.

5 Administering Oracle Database

This chapter describes how to administer Oracle Database 11*g* Release 2 for BS2000/OSD.

Common administration tasks are described in the following sections:

- Using the SQL*Plus Utility
- Startup and Parameter Files
- Remote Startup of a Database Instance
- Automatic Diagnostic Repository
- Checking the Integrity of the Physical Data Structure

5.1 Using the SQL*Plus Utility

The following topics are described in this section:

- Starting the SQL*Plus Utility in BS2000
- Calling SQL*Plus from a Procedure
- Using SQL*Plus in the POSIX shell
- Running System Commands from SQL*Plus
- Using an Editor in SQL*Plus
- Customizing SQL*Plus Profiles

5.1.1 Starting the SQL*Plus Utility in BS2000

To start SQL*Plus, enter the following:

```
/START-PROGRAM $ORAC1120.SQLPLUS
```

When you are prompted for parameters, enter `/NOLOG`:

```
* /NOLOG
```

This prevents SQL*Plus from prompting you for user name and password. Later you can explicitly connect to the database. For example:

```
SQL> CONNECT / AS SYSDBA
```

For more ways to start SQL*Plus, refer to "Running SQL*Plus" in *Oracle Database User's Guide for Fujitsu BS2000/OSD*.

5.1.2 Calling SQL*Plus from a Procedure

Set Task Switch 1 to on (/MODIFY-JOB-SWITCHES ON=1). This forces SQL*Plus to read in data from the procedure, rather than prompt you at the terminal.

5.1.3 Using SQL*Plus in the POSIX shell

Before starting SQL*Plus in the POSIX shell, you must set the environment variable ORACLE_HOME and extend the POSIX variable PATH by the path to the Oracle bin directory $ORACLE_HOME/bin. For example:

```
$ ORACLE_HOME=/u01/app/orac1120/product/dbhome_1
$ export ORACLE_HOME
$ PATH=$ORACLE_HOME/bin:$PATH
$ export PATH
```

Alternatively, you can process the profile oracle_home_path/.profile.oracle, which is created during the installation of the Oracle software under POSIX. This profile sets and expands the most important variables like ORACLE_HOME and PATH. Process the profile as follows:

```
$ . /u01/app/orac1120/product/dbhome_1/.profile.oracle
```

If you want to start SQL*Plus for a specific Oracle instance, then you must also set the variable ORACLE_SID. For starting an Oracle server process, the BS2000 variable BGJPAR is required. This variable is not set by running .profile.oracle.

If you want to start a background job with special job parameters, for example, a job should be assigned to a special JOB-CLASS, then ensure that the variable BGJPAR is set in your environment.

```
$ ORACLE_SID=orcl
$ export ORACLE_SID
$ BGJPAR='START=SOON,CPU-LIMIT=NO,J-C=JCBORA,LOGGING=*NO'
$ export BGJPAR
```

You can start SQL*Plus in the same way as on other UNIX systems with the following command:

```
$ sqlplus /nolog
$ SQL> connect / as sysdba
```

5.1.4 Running System Commands from SQL*Plus

The SQL*Plus HOST command enables you to enter a BS2000 command, while you are logged on to SQL*Plus.

Keep the following points in mind when using the HOST command:

- If you enter the HOST command without any BS2000 command, then it takes you to the command level. To return to SQL*Plus, you must use the RESUME command in BS2000 or the exit command in the POSIX subshell.
- If you enter the HOST command with a system command in the BS2000 environment, then you can only run BS2000 commands. In the POSIX shell, you can run shell commands and BS2000 commands by using the shell command bs2cmd.

5.1.5 Using an Editor in SQL*Plus

Start a text editor in SQL*Plus with the EDIT command, if you want to edit an SQL statement.

If you run SQL*Plus in the *native* BS2000 environment, then the only editor supported is BS2000 EDT.

If you run SQL*Plus in a POSIX shell, then the default editor depends on the terminal connected with your POSIX session. If the POSIX shell is started on a blockmode terminal, then the default editor in SQL*Plus is set to edtu. If the POSIX shell is started by a remote X-client through rlogin or ssh using a xterm terminal, then the default editor in SQL*Plus is set to vi.

SQL*Plus provides the opportunity to define a preferred text editor with the DEFINE _EDITOR command. In the BS2000 environment, SQL*Plus accepts any editor you specify in the DEFINE _EDITOR command, but the called editor is always the EDT. In the POSIX environment you can define a preferred editor. For example, if you want to define the editor, used by the EDIT command, to be the POSIX editor edtu, then enter the following command in SQL*Plus:

```
DEFINE _EDITOR = edtu
```

Note:

- The editor in the BS2000 environment is always the EDT.
- The editor vi does not work on blockmode terminals.
- The editor edtu does not work on xterm terminals.

5.1.6 Customizing SQL*Plus Profiles

The DBA can update the global SQL*Plus profile file, $ORAC1120.SQLPLUS.ADMIN.GLOGIN.SQL, which is run when a user logs in to SQL*Plus. This file is run before the user's local LOGIN.SQL and is provided to enable sites to set up several defaults useful to all users. You can place any SQL and SQL*Plus statement in GLOGIN.SQL.

See Also: *SQL*Plus User's Guide and Reference* for more information about customizing SQL*PLUS profiles

5.2 Startup and Parameter Files

Oracle uses two parameter files when starting the database:

1. The ORAENV file, the environment definition file, which contains BS2000-specific information. In the ORAENV file you identify the database to be started, or shut down. You can use this file to set configuration variables, which adapt the Oracle Database to the local operating system and application environment.
2. The initialization file INIT.ORA or the server parameter file SPFILE, which exists in all Oracle Database implementations and contains database-specific parameters.

This section describes the following:

- The Environment Definition File ORAENV
- The Initialization File INIT.ORA

- The Server Parameter File SPFILE
- Using the Correct Initialization File

5.2.1 The Environment Definition File ORAENV

The ORAENV file is identified by *sid*.P.ORAENV, where *SID* is the database identifier. The same ORAENV file must be used by SQL*Plus in BS2000 and by all background jobs. This is ensured by the installation procedures, which create the basic ORAENV file. Refer to Appendix B, "Oracle Environment Variables", for details of required and optional ORAENV variables.

If you use SQL*Plus in the POSIX shell, then the requested BS2000 parameters set in the ORAENV file must be specified. You can set the variables in the POSIX environment or use the facility to access the BS2000 ORAENV file. When specifying the *SID* in the POSIX ORAENV filename, ensure that the *SID* in the filename and the ORACLE_SID variable use exactly the same case. For example, if you create a POSIX ORAENV file with the *SID* in the filename in uppercase, then you must set the ORACLE_SID using exactly the same *SID*.

```
$ echo '$ORADATA.ORCL.P.ORAENV' > $ORACLE_HOME/dbs/oraenvORCL
$ ORACLE_SID=ORCL
$ export ORACLE_SID
$ sqlplus /nolog
$ SQL> connect / as sysdba
```

Refer to "Starting Oracle Utilities in the POSIX environment" in *Oracle Database User's Guide for Fujitsu BS2000/OSD*, for more information about how to set POSIX environment variables.

5.2.2 The Initialization File INIT.ORA

Startup requires the INIT.ORA parameter file, which contains a list of specifications for the Oracle database. These generic, that is, platform independent parameters, are used to setup the instance. Refer to *Oracle Database Administrator's Guide* and *Oracle Database Reference* for full descriptions of these parameters.

5.2.3 The Server Parameter File SPFILE

You can choose to maintain initialization parameters in a binary server parameter file. A server parameter file is initially built from a traditional text initialization parameter file using the CREATE SPFILE command. If you enter the following command:

```
CREATE SPFILE FROM PFILE;
```

where neither SPFILE name nor PFILE name is specified, then Oracle looks for a text initialization file *sid*.DBS.INIT.ORA and creates a server parameter file *sid*.DBS.SPFILE.ORA.

5.2.4 Using the Correct Initialization File

A default initialization file, called $ORAC1120.DEMO.DBS.INIT.ORA, is distributed with Oracle Database. During the database creation procedure, this file is copied to the DBA User ID and renamed, *sid*.DBS.INIT.ORA, where *sid* is the 1 to 4 character database ID you specified at the beginning of the database creation procedure.

Oracle determines the value of *SID* by retrieving the ORASID environment variable defined in the ORAENV file for the database. When you issue the STARTUP command

with no PFILE clause, Oracle locates the initialization parameter file by examining file names in the following order:

1. *sid*.DBS.SPFILE.ORA
2. DBS.SPFILE.ORA
3. *sid*.DBS.INIT.ORA

If you want to use some other initialization file, then use the argument PFILE. For example, to bring up a previously created database using an initialization file called TEST.INIT.ORA, enter the following:

```
/START-PROGRAM $ORAC1120.SQLPLUS
* /NOLOG
```

At the SQL*Plus prompt, enter:

```
SQL> CONNECT / AS SYSDBA
SQL> STARTUP PFILE=TEST.INIT.ORA
```

5.3 Remote Startup of a Database Instance

This section describes the preparations for a remote startup using SQL*Plus:

1. Usually, Oracle Database 11*g* Release 2 listener parameter file (LISTENER.ORA) does not contain a static service registration section (SID_LIST) for a database service. In case of a remote startup you must define this section for the desired database. For example:

   ```
   SID_LIST_LISTENER = (SID_LIST =
                        (SID_DESC =
                         (SID_NAME = ORCL)))
   ```

   The listener must be running on the computer where the instance is to be started. The listener must statically register the instance. If the listener does not run under the same user ID as the instance you want to start, then you must define the admissions to start a job under the user ID of the instance in the listener's ORAENV file or you must use SECOS, Fujitsu's Security Control System for BS2000/OSD. For more information refer to the Configuring the Network chapter in the Oracle Net Services section of this guide.

2. Create a password file with the Oracle utility ORAPWD under the user ID of the instance you want to administrate. For more information of how to use ORAPWD, refer to *Oracle Database Administrator's Guide*. To run the ORAPWD utility on BS2000, use the following command:

   ```
   /START-PROGRAM $ORAC1120.ORAPWD
   *file=password_file password=my_password entries=10
   ```

3. The name of the password file is taken from the parameter SSSIDPWF. So you must add this parameter to the ORAENV file of the instance you want to start:

   ```
   SSSIDPWF = password_file
   ```

4. The parameter REMOTE_LOGIN_PASSWORDFILE must be set to EXCLUSIVE in the initialization file of the instance.

   ```
   REMOTE_LOGIN_PASSWORDFILE = EXCLUSIVE
   ```

5. Execute SQL*Plus on the remote computer and connect as user `sys` to a server of the instance you want to startup. In the following example, which shows the commands for SQL*Plus on a UNIX client, we use the net service name *orcl_on_bs2000* to address the remote instance on the BS2000 computer:

```
sqlplus /nolog
SQL> connect sys@orcl_on_bs2000 as sysdba
Enter password:
password
Connected
SQL> startup
...
```

5.4 Automatic Diagnostic Repository

Automatic Diagnostic Repository (ADR) is a file-based hierarchical data store for depositing diagnostic information produced by diagnostic framework clients. The repository contains data describing incidents, traces, dumps, alert logs, health check records, SQL Trace information, and other information essential for problem diagnosis.

> **See Also:** "Managing Diagnostic Data" in *Oracle Database Administrators Guide* for more information about Automatic Diagnostic Repository

This section describes the following:

- Automatic Diagnostic Repository Directories and Files
- ADR Command Interpreter

5.4.1 Automatic Diagnostic Repository Directories and Files

Automatic Diagnostic Repository (ADR) is a directory structure that is stored outside of the database. It is therefore available for problem diagnosis when the database is down.

The directories and files of the Automatic Diagnostic Repository are stored in the POSIX file system.

> **See Also:** *Oracle Database Administrator's Guide* for information about the directory structure

The ADR root directory is known as ADR base. Its location is set by the `DIAGNOSTIC_DEST` initialization parameter. For example:

```
DIAGNOSTIC_DEST=/u01/app/orac1120/oradata/adr
```

If this parameter is omitted or left null, then the database sets `DIAGNOSTIC_DEST` upon startup as follows:

- If environment variable `ORACLE_BASE` is set, then `DIAGNOSTIC_DEST` is set to the directory designated by `ORACLE_BASE`.
- If environment variable `ORACLE_BASE` is not set, then `DIAGNOSTIC_DEST` is set to `ORACLE_HOME/log`.

Within ADR base, there can be multiple ADR homes, where each ADR home is the `root` directory for all diagnostic data, such as traces, dumps, the alert log, and so on, for a particular instance of a particular Oracle product or component.

Oracle Net Services also store diagnostic data in the ADR. The location for diagnostic information for Oracle Net Services is set by the parameters `ADR_BASE` and `ADR_BASE_`*`listener_name`*. These are set in the Oracle Net Services parameter files `sqlnet.ora` and `listener.ora`.

> **See Also:** "Troubleshooting Oracle Net Services" in *Oracle Net Services Administrator's Guide* for more information about diagnostic information for Oracle Net Services

You can read the text files of the Automatic Diagnostic Repository with text editors such as `vi`, `edtu`, or with POSIX shell commands such as `cat`, `more`.

You can investigate the ADR with the utility ADRCI.

5.4.2 ADR Command Interpreter

ADR Command Interpreter (ADRCI) is a utility that enables you to investigate problems, view health check reports, and package first-failure diagnostic data within a command-line environment. You can then upload the package to Oracle Support. ADRCI also enables you to view the names of the trace files in the ADR, and to view the alert log with XML tags stripped, with and without content filtering.

> **See Also:** "ADRCI: ADR Command Interpreter" in *Oracle Database Utilities* for more information about ADRCI

Start the command-line utility ADRCI in the POSIX shell. Before starting ADRCI, set the environment variable `ORACLE_HOME` and extend the environment variable `PATH` by the path name of the Oracle directory `oracle_home_path/bin`. Additional environment variables, such as `ORACLE_SID`, are not required. Refer to "Starting Oracle Utilities in the POSIX environment" in *Oracle Database User's Guide for Fujitsu BS2000/OSD* for more information about Oracle Database utilities.

After setting these environment variables, start ADRCI in the POSIX shell by entering `adrci` after the POSIX shell command prompt.

> **Note:** ADRCI cannot be started in the normal BS2000 environment. ADRCI must be started in the POSIX shell.

Display the current ADR base with the ADRCI `show base` command. The current ADR home can be displayed with the `show homes` command.

Set ADR base with the `set base` command. ADR home can be set with the `set home` command.

The command `show alert` shows the contents of the alert log in a text editor.

If you use a blockmode terminal, then the default editor for ADRCI on Fujitsu BS2000/OSD is `edtu`. If you use an xterm terminal after having logged in to POSIX through `rlogin` or `ssh`, then the default editor for ADRCI is `vi`.

You can either select the preferred text editor by setting the environment variable `EDITOR` before starting ADRCI or specify your preferred text editor within ADRCI with the ADRCI `set editor` command.

> **Note:** The editor *vi* does not work on blockmode terminals and the editor *edtu* does not work on xterm terminals.

With ADRCI you can invoke Incident Packaging Service (IPS) to create packages for incidents with the commands `ips create package` and `ips generate package`. The resulting zip file can be uploaded to Oracle Support.

5.5 Checking the Integrity of the Physical Data Structure

To check the data-structure integrity of offline databases, use the `DBVERIFY` command-line utility. To start `DBVERIFY` enter the following command:

```
/START-PROGRAM $ORAC1120.DBV
```

You can now enter the command, for example:

```
file=ora11.dbs.database1.dbf blocksize=4096 feedback=100
```

> **See Also:** *Oracle Database Administrator's Guide* and *Oracle Database Utilities* for more information about the `DBVERIFY` program

6

Backing Up and Recovering a Database

This chapter supplements the generic Oracle Database documentation set with information about backup and recovery.

Refer to the following Oracle manuals for detailed information about database backup and recovery:

- *Oracle Database Concepts*
- *Oracle Database Administrator's Guide*
- *Oracle Database Backup and Recovery User's Guide*

You can choose among many methods and Oracle tools for backup and recovery. You may use the Import and Export Utilities for logical backup and recovery. For physical backup and recovery you may use Recovery Manager (RMAN) or operating system utilities.

This chapter describes some BS2000 specific issues if you apply user-managed backup and recovery with SQL*Plus and BS2000 utilities.

Refer to Section 6.1.2, "Online Backup" for information about RMAN on BS2000.

6.1 Backing Up an Oracle Database

You can use one of the following methods to back up an Oracle Database.

- Using BS2000 Utilities to Back Up an Oracle Database
- Online Backup

6.1.1 Using BS2000 Utilities to Back Up an Oracle Database

You can back up an Oracle database using BS2000 operating system utilities (for example, `ARCHIVE` or the `/COPY-FILE` command).

Use the following steps to back up an Oracle database:

1. While the database is running, collect the names of all files, which make up Oracle database. You can determine the names of the log and database files by entering the following commands:

```
/START-PROGRAM $ORAC1120.SQLPLUS
* /NOLOG
SQL> CONNECT / AS SYSDBA
SQL> SELECT * FROM V$DATAFILE;
SQL> SELECT * FROM V$LOGFILE;
```

2. To ensure that all Oracle database files are synchronized at the time of the backup, shut down Oracle database using SQL*Plus.
3. Back up all database files and log files using the BS2000 ARCHIVE utility or the BS2000 /COPY-FILE command. You should always back up all files at the same time.
4. Restart Oracle Database using SQL*Plus.

6.1.2 Online Backup

You can perform an online backup of the database or individual tablespaces by using either:

- the BS2000 ARCHIVE utility together with Oracle Database INSTALL.C.OPNBACK utility
- the BS2000 PERCON utility

The ARCHIVE method is faster, and is described in this section.

Before you can perform an online (hot) backup of individual tablespaces, you must ensure that the ARCHIVE utility can back up open files.

The following BS2000 command ensures that ARCHIVE can back up all open files:

```
/START-PROGRAM  $ORAC1120.INSTALL.C.OPNBACK
*filename
```

The INSTALL.C.OPNBACK utility calls the BS2000 macro CATAL, which sets the OPNBACK file attribute to YES. For the CATAL macro to work the database must be shut down or the tablespace in question must be offline. You enter this command once for each file. For example, before adding it to a tablespace, not on the occasion of each backup.

You must never back up database files online without first setting the tablespace to backup mode. If you do not follow this step, then the resulting backup files are inconsistent. To perform an online backup of individual tablespaces or data files, use the following procedure:

1. Enter the following command:

   ```
   SQL> ALTER TABLESPACE name BEGIN BACKUP;
   ```

2. Back up the files of the tablespace using the BS2000 utility ARCHIVE utility. Ensure that the OLS parameter of ARCHIVE is set to YES.
3. Enter the following command:

   ```
   SQL> ALTER TABLESPACE name END BACKUP;
   ```

Note: The preceding SQL*Plus commands operate on tablespaces, while the ARCHIVE utility operates on data files.

6.2 Recovering Databases

An Oracle Database can be restored offline from backups, using the following steps:

1. Copy all the database files and the log files from the backup. You may use the BS2000 ARCHIVE utility or the BS2000 /COPY-FILE command. Files must be restored with their original name.

While Oracle database is running, the names of all files which it comprises can be determined by querying the table V$DATAFILE. Enter the following command when the SQL prompt is displayed:

```
SQL> select file#,name FROM V$DATAFILE;
```

The following is an example of the result that is displayed:

```
FILE#NAME
------------------------------
1    :pvs:$dbauserid.sid.DBS.DATABASE1.DBF
2    :pvs:$dbauserid.sid.DBS.DATABASE2.DBF

2 ROWS SELECTED.
```

You can determine the name of the log files in a similar way:

```
SQL> SELECT GROUP#,MEMBER FROM V$LOGFILE;
```

The following is an example of the result that is displayed:

```
GROUP#MEMBER
------------------------------
1     :pvs:$dbauserid.sid.DBS.LOG1.DBF
2     :pvs:$dbauserid.sid.DBS.LOG2.DBF

2 ROWS SELECTED.
```

2. Under the DBA user ID, ensure that the ORA*SID* environment variable identifies the Oracle Database, which is to be restored.
3. Use the SQL*Plus STARTUP command to start the Oracle database.

6.3 Recovery Manager

On BS2000/OSD, Recovery Manager does not support tapes. Disks are the only backup media.

As a workaround, you could use the Recovery Manager output as a first level storage to be 'migrated' by BS2000 subsystem HSMS (Hierarchical Storage Management System) to tapes. However, it is the administrator's responsibility to care for cooperation of the two systems.

Recovery Manager must use an Oracle Net Services connection. For this purpose you can:

- use a Bequeath Server (refer to Chapter 9, "Oracle Net Services")
- create a password file, start a listener, and connect through Oracle Net Services

The following is an example of a Recovery Manager command:

```
/START-PROG $ORAC1120.RMAN
*target "dba1/dba1@i1" catalog "dba2/dba2@i2" cmdfile "b.dat" log "b.log"
```

7

Tuning Oracle Database

Careful tuning can dramatically increase the speed of an Oracle database. This chapter describes how you can monitor and tune the system for optimal performance of Oracle database. For a general discussion of performance and tuning issues, refer to *Oracle Database Administrator's Guide* and *Oracle Database Performance Tuning Guide*.

This chapter covers BS2000 specific tuning parameters.

You should always tune Oracle database. In addition, you may need to fine tune or reconfigure the BS2000/OSD operating system to achieve optimal performance, or to support more users than the basic configuration enables.

7.1 BS2000/OSD-Specific Parameters

This section describes `ORAENV` environment variables for BS2000/OSD-specific Oracle Database performance tuning.

7.1.1 Task Priority

Apart from generic or BS2000/OSD-specific Oracle Database tuning options, the BS2000/OSD priority of Oracle Database tasks at run-time, and their run-time priority balance, can have a lot of influence on the overall throughput and the time delays experienced.

Initially, all foreground (network server) and background tasks have the same run-time priority, as specified by the `ORAENV` environment variable, `BGJPAR`. Server tasks automatically become TP tasks, and the background tasks also switches to TP mode, if the `JOIN` entry permits this.

Both before and while changing run-time priorities, you should use SM2 to investigate task behavior and possible bottlenecks. There is no simple rule that says that certain tasks always have a high priority.

8

openUTM Product Set

This chapter describes how to use the BS2000/OSD transaction monitor openUTM for coordinated interoperation with Oracle Database 11*g* Release 2 (11.2). The following areas are covered:

- Operation of Oracle Database Using openUTM Programs
- Distributed openUTM Files
- Installing
- Developing an Oracle Database/openUTM Application
- Troubleshooting

8.1 Operation of Oracle Database Using openUTM Programs

Universal Transaction Monitor (openUTM) controls the execution of user programs that can be used from a large number of terminals at the same time.

An openUTM application consists of a structured sequence of processing stages that are supplied with access rights for the specific user. These stages, in turn, consist of openUTM transactions that are carried out either in their entirety, or not at all.

If several users are working under openUTM at the same time, then simultaneous access to the shared database is also usually required. The database/data communications system (DB/DC system), Oracle Database/openUTM, synchronizes access by openUTM applications to Oracle Database, and ensures that the database remains in a consistent state. In the event of system failure, the DB/DC system performs an automatic recovery, which ensures that the database remains in a consistent state.

Synchronization of Oracle and openUTM is done through the XA interface. The XA interface is an X/Open interface for the coordination between database systems and transaction monitors. Refer to Developing Applications with Oracle XA chapter in *Oracle Database Advanced Application Developer's Guide* for a description of the concepts of the XA interface.

8.2 Distributed openUTM Files

When you install Oracle Database, as described in Chapter 3, "Oracle Database Installation and Deinstallation", the openUTM related software of the Oracle Database software is installed. The distributed openUTM files comprise of:

- `XAO.LIB`

  This file contains the connection module for the XA interface.

- The following files provide examples of procedures and programs:

```
UTM.DEMO.P.COMPILE.C
UTM.DEMO.P.COMPILE.COBOL
UTM.DEMO.P.KDCDEF
UTM.DEMO.P.KDCROOT
UTM.DEMO.P.PROBIND
UTM.DEMO.P.PROSTRT
UTM.DEMO.CSELEMP.PC
UTM.DEMO.SELDEP.PCO
UTM.DEMO.SELEMP.PCO
UTM.DEMO.UPDEMP.PCO
UTM.DEMO.ERRSQL.C
UTM.DEMO.ERRTXT.C
```

8.3 Installing

Perform the following step to install after studying the *Oracle Database Advanced Application Developer's Guide*:

- Grant the `SELECT` privilege to the `DBA_PENDING_TRANSACTIONS` table for all openUTM users connecting to the Oracle Database. Use the following example to grant the `SELECT` privilege to user `scott`:

  ```
  grant select on DBA_PENDING_TRANSACTIONS to scott;
  ```

  The openUTM users are identified in the Open String with the Item `Acc`. Refer to Defining an Open String section on page 8-4 in this chapter.

8.4 Developing an Oracle Database/openUTM Application

Oracle Database 11*g* on BS2000 supports openUTM V6.0 or higher. openUTM supports the XA interface. Oracle Database 11*g* on BS2000 coordinates with openUTM through this XA interface.

The steps involved in developing an Oracle Database application for coordinated inter-operation with openUTM are described in this section. The main steps are as follows:

1. Building the openUTM program units
2. Defining the configuration
3. Translating the KDCROOT table module and openUTM program units
4. Linking the openUTM application program
5. Starting the openUTM application

In addition, this section also describes how you define open strings and how you use precompilers with the Oracle XA library.

1. Building the openUTM program units:

   (refer to the openUTM manual *Programming Applications with KDCS for COBOL, C, and C++*, and the *Oracle Database User's Guide for Fujitsu BS2000/OSD*)

2. Defining the configuration:

   (refer to the openUTM manuals *Generating Applications* and *Administering Applications*)

An Oracle Database/openUTM application requires the following information for execution:

- Information about the application
- Username/password with access protection
- Information about the terminal and communication partners
- Information about the transaction codes

These properties collectively form the configuration, which is stored in the KDCFILE file. The configuration definition is carried out by the KDCDEF utility.

This section gives the descriptions for three commands that are important for connecting to the Oracle database. They are:

- DATABASE

  When the Oracle Database/openUTM application is generated, you must specify that openUTM communicates with the Oracle Database. Enter the following command to specify openUTM communication with the database:

  ```
  DATABASE TYPE=XA,ENTRY=XAOSWD
  ```

  where TYPE=XA specifies the use of the XA interface and ENTRY=XAOSWD specifies the name of the XA switch for the Oracle database (for dynamic registration).

- OPTION

  If you specify the corresponding GEN operand in the OPTION command, then the KDCDEF utility also produces the source-code for the KDCROOT table module. The syntax of OPTION is as follows:

  ```
  OPTION [DATA=filename][,GEN={KDCFILE|ROOTSRC|NO|ALL}]
  [,ROOTSRC=filename][,SHARETAB=filename]
  [,TEST={N[o]|Y[ES]}]
  ```

- MAX

  Another important operand is APPLIMODE, which is specified in the MAX command. This determines restart behavior after a system failure. The syntax of MAX is as follows:

  ```
  MAX APPLINAME=name[,APPLIMODE={S[ECURE]|F[AST]}]
  [,ASYNTASKS=number][...]
  ```

  APPLIMODE=SECURE means that openUTM continues after an application malfunction with a coordinated warm-start of the openUTM application and the Oracle database.

  If you specify APPLIMODE=FAST, then no openUTM application restart is executed, as openUTM stores no restart information. In the event of an error, the application starts from scratch. Transactions that are still open after an openUTM-application malfunction are rolled back automatically.

See the UTM.DEMO.P.KDCDEF file for an example procedure for building the KDCFILE and the KDCROOT table module.

3. Translating the KDCROOT table module and openUTM program units:

   The source of the KDCROOT table module should be compiled with the BS2000 Assembler and the openUTM program units should be compiled with the

corresponding programming language compilers. See the example procedure UTM.DEMO.P.KDCROOT for the compilation of the KDCROOT table module.

4. Linking the openUTM application program:

   The openUTM application program is produced by linking the KDCROOT table module with the openUTM program units.

   You must include the stub module XAOSTUB:

   ```
   INC-MOD LIB=ORAUID.XAO.LIB,ELEM=XAOSTUB
   ```

   Note: Instead of writing the binding procedure, you should use the example procedure UTM.DEMO.P.PROBIND and apply modifications when needed.

   If you must write your own binding procedure, then study the example carefully before writing one.

5. Starting the openUTM application:

   An example procedure for starting the openUTM application can be found in the file UTM.DEMO.P.PROSTRT.

   When starting the openUTM application, you must specify the start parameters for openUTM, as well as for the Oracle Database.

   The openUTM start parameters are described in the openUTM manual *Using openUTM Applications under BS2000/OSD*.

   The start parameter for using the XA interface for coordinated inter-operation with Oracle Database 10*g* is:

   ```
   .RMXA RM="Oracle_XA",OS="<ORACLE open string>"
   ```

8.4.1 Defining an Open String

This section describes how to construct an open string. The transaction monitor uses this string to open the database. The maximum number of characters in an open string is 256, and the maximum number of open strings is 8. Construct the string as follows:

```
Oracle_XA{+required_fields...}[+optional_fields...]
```

where the *required_fields* are:

- Acc=*P/user/access_info*
- SesTm=*session_time_limit*

and the *optional_fields* are:

- DB=*db_name*
- MaxCur=*maximum_no_of_open_cursors*
- SqlNet=*connect_string*
- DbgFl=*value_from_1_to_15*

Notes:

Remember the following:

- You can enter the required fields and optional fields in any order when constructing the open string.
- All field names are case-insensitive, although their values may or may not be case-sensitive depending on the system.
- You may not use the "+" character as part of the actual open string.

8.4.1.1 Required Fields

The required fields for the open string are:

Item	Meaning
`Acc`	Specifies user access information.
`P`	Indicates that explicit user and password information is provided.
`user`	A valid Oracle Database username.
`access_info`	The corresponding current password.

For example, `Acc=P/scott/tiger` indicates that user and password information is provided. In this case, the user is `scott` and the password is `tiger`.

For the correct process, ensure that `scott` has the `SELECT` privilege on the `DBA_PENDING_TRANSACTIONS` table.

Item	Meaning
`SesTm`	Specifies the maximum amount of time a transaction can be inactive before it is automatically deleted by the system.
`session_time_limit`	This value should correspond to what you require as a maximum time from initiation of a global transaction and the completed commit or rollback of that transaction.

8.4.1.2 Optional Fields

Optional fields for the open string are described in the following table:

Item	Meaning
`DB`	Specifies the database name.
`db_name`	Indicates the name used in Oracle Database precompilers to identify the database. Application programs that use only the default database for the Oracle Database precompiler, that is, do not use the AT clause in their SQL statements, should omit the `DB=`*`db_name`* clause in the open string. **Note:** This default database is represented in the `ORAENV` file by `ORASID`. Applications that use explicitly–named databases should indicate that database name in their `DB=`*`db_name`* field. For example, `DB=payroll` indicates that the database name is payroll and that the application server program uses that name in AT clauses.

For more information about precompilers, specifically Pro*C, refer to the section Using Precompilers with openUTM later in this chapter.

Item	Meaning
`MaxCur`	Specifies the number of cursors to be allocated when the database is opened. It serves the same purpose as the precompiler option `maxopencursors`.
`maximum_no_of_open_cursors`	Indicates the number of open cursors. The default is 10.

For example, `MaxCur=5` indicates that the process should try to keep five open cursors cached.

For more information about `maxopencursors`, refer to the *Oracle Database Programmer's Guide to the Oracle Precompilers*.

Item	Meaning
`SqlNet`	Specifies the SQL*Net connection string.
`connect_string`	Indicates the string to be used to log onto the system. This can be any supported Oracle Net Services connect string.

For example:

`SqlNet=MADRID_FINANCE` indicates an entry in `TNSNAMES.ORA` referencing a protocol, a host, and a portnumber. For more information, refer to Chapter 9, "Oracle Net Services" in this book.

Item	Meaning
`DbgFl`	Specifies if debugging should be enabled (debug flag). For more information refer to Debugging in the subsequent section in this chapter.

8.4.1.3 Examples

This section contains examples of open strings using the preceding information.

> **Note:** If the string is longer than one line, then refer to the openUTM documentation for information about how to split up the string information.

For bequeath protocol:

```
Oracle_XA+Acc=P/scott/tiger+SesTm=0+DbgFl=15
```

For other protocols:

```
Oracle_XA+SqlNet=MADRID_FINANCE+Acc=P/scott/tiger+SesTm=0
Oracle_XA+DB=finance+SqlNet=MADRID_FINANCE+Acc=P/scott/tiger
+SesTM=0
```

The optional fields `LogDir`, `Loose_Coupling`, `SesWT`, and `Threads` are not supported.

For more information about the fields in the open string refer to the *Developing Applications with Oracle XA* section in *Oracle Database Advanced Application Developer's Guide*.

8.4.2 Using Precompilers with openUTM

You can choose from two options when interfacing with precompilers:

- Using precompilers with the default database
- Using precompilers with a named database

You should run all precompiler programs with the option `release_cursor` set to `no`. Precompiler programs may be written in C or COBOL. In the following examples, the precompiler Pro*C is used.

8.4.2.1 Using Pro*C with the Default Database

To interface to Pro*C with the default database, ensure that the `DB=`*`db_name`* field used in the open string is not present. The absence of this field indicates the default connection as defined in the `ORAENV` file, and only one default connection is allowed for each process.

The following is an example of an open string identifying a default Pro*C connection:

```
Oracle_XA+SqlNet=MADRID_FINANCE+Acc=P/scott/tiger+SesTm=0
```

Here, `DB=`*`db_name`* is absent, indicating an empty database identifier string.

The following is the syntax of a select statement:

```
EXEC SQL SELECT ENAME FROM EMP;
```

8.4.2.2 Using Pro*C with a Named Database

To interface to Pro*C with a named database, include the `DB=`*`db_name`* field in the open string. Any database you refer to must reference the same *`db_name`* specified in the corresponding open string.

An application may include the default database, as well as one or more named databases, as shown in the following examples.

For example, suppose you want to update an employee's salary in one database, the department number `deptno` in another, and the manager information in a third database. You would configure the following open strings in the transaction manager:

```
Oracle_XA+SqlNet=MADRID_FINANCE1+Acc=P/scott/tiger+SesTm=0
Oracle_XA+DB=MANAGERS+SqlNet=MADRID_FINANCE2+
Acc=P/scott/tiger+SesTm=0
Oracle_XA+DB=PAYROLL+SqlNet=MADRID_FINANCE3+
Acc=P/scott/tiger+SesTm=0
```

There is no `DB=`*`db_name`* field in the first open string.

In the application program, you would enter declarations such as:

```
EXEC SQL DECLARE PAYROLL DATABASE;
EXEC SQL DECLARE MANAGERS DATABASE;
```

Again, the default connection corresponding to the first open string that does not contain the *`db_name`* field, does not require a declaration.

When doing the update, enter statements similar to the following:

```
EXEC SQL AT PAYROLL update emp set sal=4500 where empno=7788;
EXEC SQL AT MANAGERS update emp set mgr=7566 where empno=7788;
EXEC SQL update emp set deptno=30 where empno=7788;
```

There is no AT clause in the last statement because it refers to the default database.

You can use a character host variable in the AT clause, as the following example shows:

```
EXEC SQL BEGIN DECLARE SECTION;
db_name1 CHARACTER(10);
db_name2 CHARACTER(10)
EXEC SQL END DECLARE SECTION;
.
.
set db_name1 = 'PAYROLL'
set db_name2 = 'MANAGERS'
.
.
EXEC SQL AT :db_name1 UPDATE...
EXEC SQL AT :db_name2 UPDATE...
```

For more information, refer to the respective sections in the *Pro*COBOL Programmer's Guide* and *Pro*C/C++ Programmer's Guide* that discusses concurrent logons.

Note:

- Application servers must not create Oracle database connections of their own. Therefore, an openUTM user is not allowed to issue `CONNECT` statements within an openUTM program. Any work performed by them would be outside the global transaction, and may confuse the connection information given by openUTM.
- SQL calls must not occur in the openUTM start exit routine, however may occur in the conversation exit routine (`Vorgangs-Exit`)

8.5 Troubleshooting

This section discusses how to recover data if there are problems or a system failure. Both trace files and recovering pending transactions are discussed in the following sections.

8.5.1 Trace Files

The Oracle XA library logs any error and tracing information to its trace file. This information is useful in supplementing the XA error codes. For example, it can indicate whether an open failure is caused by an incorrect open string, failure to find the Oracle Database instance, or a login authorization failure. The name of the trace file is:

`ORAXALOG.`*`pid-db_name-date`*`.TRC`

where

`pid` is the process identifier (TSN)

`db_name` is the database name you specified in the open string field `DB=`*`db_name`*

`date` is the date when the trace file is created

8.5.1.1 Trace File Examples

Examples of two types of trace files are discussed in this section.

The following example shows a trace file for an application's task '1234' that was opened on April 2nd 1999. The DB field for this application was not specified in the open string when the resource manager was opened

```
ORAXALOG.1234-NULL-990402.TRC
```

The following example shows a trace file that was created on December 15th 1998 by task 5678. The DB field was specified as FINANCE in the open string when the resource manager was opened.

```
ORAXALOG.5678-FINANCE-981215.TRC
```

Each entry in the trace file contains information that looks like this:

```
1032.2: xa_switch rtn ORA-22
```

where 1032 is the time when the information is logged, 2 is the resource manager identifier, xa_switch is the module name, and ORA-22 is the returned Oracle database information.

8.5.2 Debugging

You can specify the DbgFl (debug flag) in the open string. For more information, refer to the *Oracle XA* chapter in *Oracle Database Advanced Application Developer's Guide*.

Depending on the debugging level (low:DbgFl=1,high:DbgFl=15) you can get more or less debug entries in the trace file ORAXALOG.*pid-db_name-date*.TRC (refer to the preceding section).

8.5.3 In-Doubt or Pending Transactions

In-doubt or pending transactions are transactions that have been prepared but not yet committed to the database. Generally, openUTM resolves any failure and recovery of any in-doubt or pending transaction. However, the Database Administrator may have to override an in-doubt transaction in working with UTM-F, that is, APPLIMODE=FAST, for example when the in-doubt transaction is:

- Locking data that is required by other transactions
- Not resolved in a reasonable amount of time

> **Note:** Overriding in-doubt transactions can cause inconsistency between openUTM and the database. For example, if the DB transaction is committed by the Database Administrator and the openUTM application rolls back the transaction in the warm-start phase, then the Oracle Database cannot roll this committed transaction back, therefore, causing an inconsistency.

8.5.4 Oracle Database SYS Account Tables

There are four tables under the Oracle Database SYS account that contain transactions generated by regular Oracle Database applications and Oracle Database/openUTM applications. These are as follows:

- DBA_2PC_PENDING
- DBA_2PC_NEIGHBORS
- DBA_PENDING_TRANSACTIONS

- V$GLOBAL_TRANSACTION

> **Note:** For detailed information about how to use these tables, refer to the sections in the *Oracle Database Administrator's Guide* that discuss failures during two-phase commit and manually overriding in-doubt transactions.

For transactions generated by Oracle Database/openUTM applications, the following column information applies specifically to the DBA_2PC_NEIGHBORS table:

- The DBID column is always xa_orcl.
- The DBUSER_OWNER column is always *db_name*xa.oracle.com.

Remember that the *db_name* is always specified as DB=*db_name* in the open string. If you do not specify this field in the open string, then the value of this column is NULLxa.oracle.com for transactions that are generated by Oracle Database/openUTM applications.

For example, you could use the following sample SQL statement to find out more information about in-doubt transactions that are generated by Oracle Database/openUTM applications.

```
SELECT * FROM DBA_2PC_PENDING p, DBA_2PC_NEIGHBORS n
WHERE p.LOCAL_TRAN_ID = n.LOCAL_TRAN_ID AND n.DBID = 'xa_orcl';
```

9

Oracle Net Services

This chapter describes Oracle Net Services and it's implementation in the BS2000/OSD environment. It supplements the *Oracle Database Net Services Administrator's Guide* with BS2000/OSD-specific information about the following topics:

- Introducing Oracle Net Services
- Shared Server Architecture
- Oracle Advanced Security
- Configuring the Network
- Troubleshooting Oracle Net Services

9.1 Introducing Oracle Net Services

Oracle Net Services supports network communication between a client application and a remote or local database running on a variety of operating systems.

Oracle Net Services allows the database servers and the client applications, or servers acting as clients, to run on separate systems, and provides a means for moving data between the nodes on a network. For example, a UNIX or Windows user can run applications that access and manipulate data in a remote Oracle database running on a BS2000 system

Oracle Net Services is also used for Inter Process Communication if clients and database are running on the same system.

9.1.1 IPC Protocol Support

This section introduces Oracle's Interprocess Communication (IPC) protocol support for inter-process calls. It is used to map the functionality of IPC to Oracle's Net Foundation Layer. The IPC protocol is supported for native BS2000 only. Oracle Utilities and products running in the POSIX shell cannot use the IPC protocol.

9.1.1.1 Overview of IPC

On BS2000 systems, the IPC protocol is used for local inter-process communication. The Oracle Protocol Support for IPC uses the ISO functionality of the BS2000 sockets.

The client process initiates its IPC connection with the remote process by specifying a `KEY` that describes the listening process. Once the connection is established, the two communicating processes send and receive data through a continuous byte stream.

9.1.1.2 Using the IPC Protocol

The IPC protocol allows applications to integrate with the Inter Process Communication method on a local host. The following is the syntax for using IPC protocol:

```
(ADDRESS=
   (PROTOCOL=IPC)
   (KEY=alphanumeric)
)
```

where

`PROTOCOL` specifies the supported protocol. For IPC, the value is "IPC".

`KEY` specifies the listen endpoint. A string of at most 32 characters: [a...z], [A...Z], [0...9], '.', '-', '_', '$'

The following is an example of an IPC ADDRESS that specifies a server on a local host:

```
(ADDRESS=
   (PROTOCOL=IPC)
   (KEY=ORCL)
)
```

Note: The IPC protocol is not supported in the POSIX shell. If the IPC protocol is specified by a utility or user application running in the POSIX shell, then Oracle Net Services avoids using the IPC protocol with the following error message:

```
TNS-12557: TNS:protocol adapter not loadable
```

9.1.2 TCP/IP Protocol Support

This section introduces Oracle's TCP/IP protocol support, which is used to map the functionality within TCP/IP to Oracle's Net Foundation Layer.

9.1.2.1 Overview of TCP/IP

TCP/IP is a family of related protocols that derives its name from two main components: the Transmission Control Protocol (TCP) and the Internet Protocol (IP). The IP component dispatches information around the network, and the TCP component assures reliable transfer of data from one point to another.

Application software sitting on top of the TCP/IP protocol views the network as a reliable two-way data transmission medium. This medium provides inter-process communication in a connection-oriented manner between pairs of processes in host computers attached to inter-connected computer networks.

The application or client process initiates its TCP/IP connection with the remote host process by specifying an address pair:

- A host IP address
- A TCP port (or entry point) on the host

Once the connection is established, the pair of communicating processes sends and receives data through a continuous byte stream. The TCP/IP protocol is supported in BS2000 and the POSIX shell.

9.1.3 Using the TCP/IP Protocol

The TCP/IP protocol support implements a standard interface that is used to resolve the equivalent communication functions between the TCP/IP protocol and Oracle's Net Foundation Layer.

After the TCP/IP protocol is installed for the particular system, you can use the TCP/IP–specific parameters with the TNS connect descriptors to identify nodes within a TCP/IP-based community.

The specific TCP/IP connection parameters are part of the `ADDRESS` keyword-value pair. The three TCP/IP–specific parameters can be entered in any order within the `ADDRESS` construct. The syntax used by Oracle's TCP/IP protocol support is:

```
(ADDRESS=
    (PROTOCOL=TCP)
    (HOST=hostname)
    (PORT=port#)
)
```

where

`PROTOCOL` specifies the supported protocol. For TCP/IP, the value is `TCP`.

`HOST` specifies the host name or the host's IP address.

`PORT` specifies the TCP/IP port number.

The following is an example of the TCP/IP ADDRESS specifying a client on the `sales-server` host:

```
(ADDRESS=
    (PROTOCOL=TCP)
    (HOST=sales-server)
    (PORT=1521)
)
```

9.1.4 Bequeath Protocol

The Bequeath technique enables clients to connect to a database without using the network listener. Oracle's Bequeath protocol internally spawns a server process for each client application. It does the same operation that a remote network listener does for the connection locally.

> **Note:** Starting with Oracle Database 10g the BEQ protocol uses the handoff technique which makes this protocol incompatible with the BEQ protocol of prior versions. As a consequence of this enhancement, Oracle Database 11*g* applications cannot directly spawn a process of a version prior to Oracle Database 10*g* and connect to it through the BEQ protocol.

9.1.4.1 Overview of the Bequeath Protocol

The Bequeath protocol

- Does not use a network listener (therefore, no listener configuration is required).
- Automatically spawns a dedicated server

- Is used for local connections where an Oracle Database client application, such as SQL*Plus, communicates with an Oracle Database instance running on the same computer.
- Only works in Dedicated Server mode. It cannot be used in a Shared Server mode.

> **Note:** If clients are running under a user ID different from the DBA user ID, then Oracle recommends using a net service name to connect through a listener to the destination database.

9.2 Shared Server Architecture

The initialization parameters that control the shared server architecture are as follows:

- LOCAL_LISTENER
- DISPATCHERS
- MAX_DISPATCHERS
- SHARED_SERVERS
- MAX_SHARED_SERVERS
- SHARED_SERVER_SESSIONS
- CIRCUITS

For detailed information about the shared server architecture, refer to the *Oracle Database Net Services Administrator's Guide*.

The shared server architecture and the dedicated server architecture can work concurrently in an instance. Provide information in the connect descriptor to indicate whether a connecting application should use the shared server or the dedicated server architecture. By default, the listener process uses the shared server architecture and if you want the application to use the dedicated server architecture instead, you must set USE_DEDICATED_SERVER=ON in the SQLNET.ORA file or specify a net_service_name with the parameter SERVER in the used naming method. The SQLNET.ORA parameter USE_DEDICATED_SERVER=ON overwrites the parameter SERVER.

The following example shows how to reference a dedicated server in a shared server configuration by using a specially defined net service name:

```
FINANCE_DED=(DESCRIPTION=
                     (ADDRESS=
                          (PROTOCOL=TCP)
                          (HOST=sales-server)
                          (PORT=1521))
                     (CONNECT_DATA=
                          (SERVICE_NAME=sales.us.acme.com)
                          (SERVER=dedicated)))
```

For more information, refer to the *Oracle Database Net Services Reference* guide.

In choosing whether to use the shared server or the dedicated server architecture, you must consider the CPU overhead versus resource allocation, such as tasks, memory and so on. In a situation where many clients need to work only occasionally with the Oracle Database, it would be best to use the shared server architecture, whereas, in a situation where just a few clients need to work with the Oracle Database regularly, it would be best to use the dedicated server architecture. Your decision may not always be as clear-cut as that in these examples. If this is the case, then you can use the

information in the following shared server dynamic tables to help you arrive at your decision:

- V$DISPATCHER
- V$QUEUE
- V$SHARED_SERVERS
- V$SHARED_SERVER_MONITOR

For more information about these tables, refer to the *Oracle Database Administrator's Guide*.

9.3 Oracle Advanced Security

With this release the data integrity and the cryptographic services of Oracle Advanced Security are supported. The Secure Sockets Layer (SSL) protocol is also supported. SSL provides authentication, encryption, and data integrity using public key infrastructure (PKI). SSL stores authentication data, such as certificates and private keys, in an Oracle Wallet.

For using either the data integrity, or the cryptographic services, or both, you must specify the appropriate parameters in the SQLNET.ORA file.

Use the following parameters to specify whether a service (example: crypto-checksumming or encryption) should be active:

```
SQLNET.CRYPTO_CHECKSUM_CLIENT
SQLNET.CRYPTO_CHECKSUM_SERVER
SQLNET.ENCRYPTION_CLIENT
SQLNET.ENCRYPTION_SERVER
```

Each of the preceding parameters defaults to REJECTED.

Each of the preceding parameters can have one of the following values:

Value	Meaning
ACCEPTED	The service is active if the other side of the connection specifies REQUESTED or REQUIRED and there is a compatible algorithm available on the other side. It is inactive otherwise.
REJECTED	The service must not be active, and the connection fails if the other side specifies REQUIRED.
REQUESTED	The service is active if the other side specifies ACCEPTED, REQUESTED, or REQUIRED and there is a compatible algorithm available on the other side. It is inactive otherwise.
REQUIRED	The service must be active, and the connection fails if the other side specifies REJECTED or if there is no compatible algorithm on the other side.

Use the following parameters to control which algorithms are made available for each service on each end of a connection:

```
SQLNET.CRYPTO_CHECKSUM_TYPES_CLIENT
SQLNET.CRYPTO_CHECKSUM_TYPES_SERVER
SQLNET.ENCRYPTION_TYPES_CLIENT
SQLNET.ENCRYPTION_TYPES_SERVER
```

The value of each of these parameters can be either a list of algorithm names in parenthesis separated by commas or a single algorithm name.

The default crypto checksum type is MD5, while the encryption defaults to all the algorithms.

Type	Values
Crypto checksum types	SHA1, MD5
Encryption types	AES256, RC4_256, AES192, 3DES168, AES128, RC4_128, 3DES112, RC4_56, DES, RC4_40, DES40

Use the parameter SQLNET.CRYPTO_SEED to specify the characters used when generating cryptographic keys. The more random the characters are, the stronger the keys are. The string should be 10-70 random characters. This is required when encryption or checksumming is turned on.

```
SQLNET.CRYPTO_SEED="qwertyuiopasdfghjkl;zxcvbnm,.s1"
```

For more information, refer to *Oracle Database Net Services Administrator's Guide*, *Oracle Database Net Services Reference*, and *Oracle Database Advanced Security Administrator's Guide*.

Example:

```
SQLNET.CRYPTO_CHECKSUM_CLIENT=required
SQLNET.CRYPTO_CHECKSUM_TYPES_CLIENT=(md5)
SQLNET.ENCRYPTION_CLIENT=accepted
SQLNET.ENCRYPTION_TYPES_CLIENT=(aes192,3des168,rc4_128)
SQLNET.CRYPTO_SEED="qwertyuiopasdfghjkl;zxcvbnm,.s1"
```

9.4 Configuring the Network

Although, you can use the Easy Connect Naming Method to connect to the database, it is recommended to configure clients for the use of service names that are easy to remember aliases for database addresses and match the address preconfigured in each system's LISTENER.ORA file. The client uses these addresses to connect to the network listener, which routes the connection request to the required service. During a connection, a client passes the service name to which it wants to connect.

LISTENER.ORA file identifies and controls the behavior of the network listener that listens for services on the system. This file includes network listener descriptors and addresses, services the listener is listening for, and various control parameters.

Client configuration is accomplished by creating a list of net service names with addresses of network destinations through the local naming parameter file TNSNAMES.ORA or an LDAP compliant directory server. Clients and database servers (that are clients of other database servers) use the net service name when making a connection.

- Using Easy Connect Naming Method
- Using the Local Naming Method
- Using the Directory Naming Method
- Configuration on the Server
- Configuration on the Client

- Testing the Configuration on the Client

9.4.1 Using Easy Connect Naming Method

The Easy Connect naming method, can be used to connect to a database without the need to configure service names in the TNSNAMES.ORA configuration file. For using the Easy Connect naming method, ensure that EZCONNECT is listed in the client's configuration file parameter for naming adaptors *names.directory_path*.

Easy Connect naming is only suitable in small and simple environments. Refer to "Using the Easy Connect Naming Method" in Oracle Database Net Services Administrator's Guide for more information about the Easy Connect naming method.

9.4.2 Using the Local Naming Method

Local naming refers to the method of resolving a service name to a network address by using information configured on each individual client in a TNSNAMES.ORA configuration file. For using the local naming ensure that TNSNAMES is listed in the client's configuration file parameter for naming adaptors *names.directory_path*.

Local naming is most appropriate for simple distributed networks with a small number of services that change infrequently.

9.4.3 Using the Directory Naming Method

Directory Naming refers to the method of resolving a service name to a network address by using a Directory Server. For using a directory server ensure that LDAP is listed in the client's configuration file parameter for naming adaptors *names.directory_path* and that the target address of the directory server is configured in the parameter file LDAP.ORA, for example:

```
# LDAP.ORA Network Configuration File: network.admin.ldap.ora
DEFAULT_ADMIN_CONTEXT = ""
DIRECTORY_SERVERS= (oid_server:389:636)
DIRECTORY_SERVER_TYPE = Your Internet Directory
```

For more information, refer to the *Oracle Database Net Services Administrator's Guide*.

9.4.4 Configuration on the Server

Follow these steps to configure the Oracle Names Method:

1. Before starting the listener, you must set up the listener's configuration file LISTENER.ORA. This file includes the addresses of the listeners and various control parameters used by the listener. For more information refer to the *Oracle Database Net Services Administrator's Guide*

   On BS2000 systems you have the chance to define additional job parameters for the Oracle Database tasks particularly a PROCESSING-ADMISSION to start a job under a user ID different from the user ID of the running job. This technique provides the benefits of using only one listener for all Oracle instances on the system. The instances need not run under the same user IDs. If you want to run only one listener on the database computer, then you must specify the following parameters in the listener's environment file *sid*.P.ORAENV:

Parameter	Meaning
BGJPAR	parameters for ENTER jobs

Parameter	Meaning
*sid*_BGJPAR	parameters for ENTER jobs identified by *SID*
*sid*_USER	the user ID under which the job should run
*user*_ACCOUNT	Account of the target user ID
*user*_PASSWORD	Password of the target user ID

The following example of an ORAENV file configured for a central listener process shows how the parameters work. The listener can share this ORAENV file with an instance, which runs under the same user ID. For a better understanding, we assume that the listener and the instances DEMO and DEM1 are running under the user ID ORACDEM1 while the instance DEM2 is running under the user ID ORACDEM2. We define the following parameters:

```
BGJPAR=J-C=JCBORA,START=IMME,CPU-LIMIT=NO,LOGGING=*NO
DEM1_BGJPAR=J-C=JCBDEM1,START=IMME,CPU-LIMIT=NO
DEM2_USER=ORACDEM2
ORACDEM2_ACCOUNT=O1234
ORACDEM2_PASSWORD=ORACLE
```

The listener always runs the same sequence to look up the parameters *sid*_BGJPAR and *sid*_USER. If no value for *sid*_BGJPAR is found, then the listener uses the value given by the parameter BGJPAR. If a user ID is given by *sid*_USER, then the listener tries to get the processing admission from the parameters user_ACCOUNT and user_PASSWORD. For the given ORAENV we get the following scenarios for the listener:

- The listener should start a server for the instance DEMO. Because the parameters DEMO_BGJPAR and DEMO_USER are not defined the listener starts the server for the instance DEMO under the user ID ORACDEM1 with the start parameters defined by BGJPAR.
- If a server for the instance DEM1 must be started, then the listener looks for the parameters DEM1_BGJPAR and DEM1_USER. In this case the parameter DEM1_BGJPAR can be evaluated, whereas, the evaluation of DEM1_USER failed because this parameter is not defined. Therefore, the listener adds the start parameters "J-C=JCBDEM1,START=IMME,CPU-LIMIT=NO" to the ENTER-PROCEDURE command and starts the job under the user ID ORACDEM1.
- Now a server for instance DEM2 must be started. The listener looks for the parameters DEM2_BGJPAR and DEM2_USER. The parameter DEM2_BGJPAR is not defined so that the listener uses the start parameters defined by BGJPAR. On the other hand the parameter DEM2_USER can be evaluated successfully and returns the value ORACDEM2. Now the listener tries to get the processing admission by evaluating the parameters ORACDEM2_ACCOUNT and ORACDEM2_PASSWORD. The listener starts the server job under the user ID ORACDEM2 with the ENTER-PROCEDURE parameters "J-C=JCBORA, START=IMME,CPU-LIMIT=NO,LOGGING=*NO".

Start the listener using the Listener Control Utility LSNRCTL:

```
/CALL-PROCEDURE sid.P.ORAENV
/START-PROGRAM $ORAC1120.LSNRCTL
```

When the enter options prompt is displayed, press **ENTER** to get to the LSNRCTL prompt. Enter the following command to start the Listener:

```
LSNRCTL> START listener-name
```

9.4.5 Configuration on the Client

Configuration of network clients involves adding or editing parameters in the client configuration file SQLNET.ORA and dependent on the used naming method, the configuration file LDAP.ORA or TNSNAMES.ORA. For more information about the configuration parameters, refer to *Oracle Database Net Services Reference*.

9.4.6 Testing the Configuration on the Client

After you have verified the network connections, you can verify the connections to the desired Oracle database systems using the TNSPING utility:

```
/CALL-PROCEDURE sid.P.ORAENV
/START-PROGRAM $ORAC1120.TNSPING
```

When the enter options prompt is displayed, enter the net service name for the database service which you have specified in the naming service. If everything works fine, then a message similar to the following is returned:

```
TNS Ping Utility for BS2000/OSD S series: Version 11.2.0.2.0 -
Production on 12-JUN-2012 12:07:40
Used parameter files: network.admin.sqlnet.ora
Used TNSNAMES adapter to resolve the alias
Attempting to contact (DESCRIPTION = (ADDRESS_LIST =
(ADDRESS = (PROTOCOL = TCP)(HOST = sales-server)(PORT = 3055)))
(CONNECT_DATA = (SERVICE_NAME = sales.us.acme.com)))
OK (40 msec)
```

For more information, refer to the *Oracle Database Net Services Administrator's Guide*.

9.5 Troubleshooting Oracle Net Services

The following is a list of error messages and steps to fix the errors:

1. Listener could not be started. LSNRCTL returns the following error message:

   ```
   LSNRCTL> start
   Starting /BS2/$ORAC1120.tnslsnr: please wait...

   TNS-12547: TNS:lost contact
   TNS-12560: TNS:protocol adapter error
   TNS-00517: Lost contact
   BS2000 Error: 145: Connection timed out
   LSNRCTL>
   ```

   - Ensure that the subsystem POSIX is up and running.
   - Ensure that the BCAM Leight Weight Resolver LWRESD is properly configured and running.

2. Listener could not open the log file.
   - Check if the listener log file, for example, NETWORK.LOG.LISTENER.LOG, is accessible and readable.
   - Verify the listener log file using the BS2000 SDF command REPAIR-DISK-FILES.
   - If you are not able to repair the listener log file, then delete the file.

3. A client reports ORA-12545

- Check the naming service if the host name returned by the listener is well known in the TCP/IP network.
- If you do not want to use the BCAM host name of the computer in the TCP/IP network, then define a sockets-host-name as described in the BCAM documentation and register this name in the name service.
- Ensure that the BCAM Leight Weight Resolver LWRESD is properly configured and running.

4. A client reports ORA-12535
   - If you use the IPC protocol, then check the Connection Timeout parameter of BCAM (use the BCSHOW command). This parameter should be set to at least 600 seconds.
5. A client reports ORA-03113
   - Check if the SQLNET.EXPIRE_TIME parameter is set for the server. If the parameter is set, then check the BCAM LETTER-TIMER using the BCSHOW command. If the LETTER-TIME is less than the SQLNET.EXPIRE_TIME, then data which are sent by the server to see if the client is running may not be read during their lifetime, which is limited by the LETTER-TIME. As a result, the client logs a broken pipe in the SQLNET.LOG file:

     ```
     ns main err code: 12547
     ns (2)  err code: 12560
     nt main err code: 517
     nt (2)  err code: 32
     nt OS   err code: 0x0040002c
     ```

     You can solve this problem by setting the LETTER-TIMER to infinite.

10 Oracle Text

This chapter describes how to install and run Oracle Text and the restrictions of this option on BS2000/OSD.

10.1 Installing Oracle Text

When you follow the procedures explained in chapter 4 about creating an Oracle Database 11*g* Release 2 database the result is not Oracle Text enabled.

If you want to install Oracle Text, then complete the following steps:

1. Start SQL*Plus. To avoid being prompted for many overflow acknowledgements on the screen set `oflow=no`:

   ```
   /tchng oflow=no
   /start-program $ORAC1120.sqlplus
   * /nolog
   connect / as sysdba
   spool catctx.log
   @$ORAC1120.ctx.admin.catctx.sql CTXSYS SYSAUX TEMP NOLOCK;
   ```

   where `CTXSYS` is the ctxsys user password, `SYSAUX` is the default tablespace for `ctxsys`, `TEMP` is the temporary tablespace for `ctxsys`, and `LOCK|NOLOCK` specifies whether the `ctxsys` user account is locked or not.

2. If you are working with US english texts, then install appropriate language-specific default preferences:

   ```
   connect CTXSYS/CTXSYS
   @$ORAC1120.ctx.admin.drdefus;
   ```

   If you are not working with US english texts, then open the `drdef*.sql` script according to the preferred language, set the attribute (refer to Restrictions of Oracle Text on BS2000/OSD), and run the script.

3. Type `exit` when finished.

10.2 Starting Oracle Text utilities

The Oracle Text loader utility, `ctxldr`, is available within the Oracle Text option. For more information about its functions and parameters, refer to *Oracle Text Reference*.

The following is an example of how to start the Oracle Text utility:

```
/start-program $ORAC1120.ctxldr
*-user scott/tiger -export -name myindex -file myfile -pk 1
```

10.3 Restrictions of Oracle Text on BS2000/OSD

Due to ASCII-EBCDIC and other dependencies, the following restrictions are inherent:

- No index themes, that is `CTX_DDL.set_attribute` (`'DEFAULT_LEXER','INDEX_THEMES','NO'`) must be set
- No INSO filters and InXight linguistx technology, which are licensed on special platforms only
- No `ctxkbtc`, which is a knowledge base utility
- `URL_DATASTORE` objects are not supported
- `FILE_DATASTORE` objects may reside on native BS2000 DMS as PAM files or on the POSIX file system.

11

External Procedures

This chapter describes how to create an environment on BS2000/OSD, where external procedure calls can operate. It complements the chapter about External Routines in the Fundamentals section of *Oracle Database Advanced Application Developer's Guide*.

11.1 Loading External Procedures

This section complements the corresponding part in *Oracle Database Advanced Application Developer's Guide*.

Follow these steps to load external procedures:

1. Set up the environment.

   An external procedure does not run in the same process and address space as the caller. Oracle creates separate processes for them to operate in a safe and secure manner. For this purpose Oracle Net Services features are used and it is the responsibility of the user to provide appropriate Oracle Net Services parameter files. In this section, we have documented an example of how it can work. For more information, refer to *Oracle Database Advanced Application Developer's Guide*, *Oracle Database Data Cartridge Developer's Guide*, and *Oracle Database Net Services* documentation set.

   The `listener.ora` file should have the following entry:

   ```
   (SID_DESC = (SID_NAME = ep_agt1)
               (ENVS = EXTPROC_DLLS=ANY)
               (ORACLE_SID = sid_of_your_database)
               (PROGRAM = EXTPROC)
   )
   ```

   The `tnsnames.ora` file should have the following entry:

   ```
   extproc_connection_data = (DESCRIPTION =(ADDRESS =(PROTOCOL=ipc)
   (KEY = sid_of_your_database)(CONNECT_DATA =(PRESENTATION=RO)
   (SID = ep_agt1)))
   ```

   Then you can start a listener.

2. Identify the 'DLL'.

   A DLL in the BS2000 environment is a BS2000 LMS library containing the functions called as external procedures. When `EXTPROC` is loaded these functions are dynamically bound to the program.

   Use the following command to identify your library to Oracle:

   ```
   CREATE LIBRARY my_c_library AS '$myuserid.my-modlib';
   ```

The external C routine has to be compiled and the generated LLM has to be stored in the LMS library. If you set the `MODULE-PROPERTIES` option `LOWER-CASE-NAMES=*NO`, then all lowercase letters in the entry names are converted to uppercase.

Furthermore, if you set the MODULE-PROPERTIES option `SPECIAL-CHARACTERS=*CONVERT-TO-DOLLAR`, then all underscores (_) in entry names are converted to dollar signs ($), which must be considered when publishing the external procedure.

3. Publishing and running external procedures does not differ from the description in *Oracle Database Advanced Application Developer's Guide*. Result messages about the execution of the external procedure can be found in a file named `L.`*`sid`*`.EXTP.SYSOUT.`*`tsn`*.

12

Java in the Database

This chapter describes BS2000-specific features for Java in the database. This chapter includes:

- Installation of a Java Enabled Database
- Database character sets and Java Encodings
- Java Demonstration Files

For more information, refer to the Oracle Java documentation set.

12.1 Installation of a Java Enabled Database

When you call `$ORAC1120.INSTALL.P.SUPER` and set the JAVA parameter to YES, you'll get a suitable `ORAENV` file (with `ORACLE_HOME` as described earlier), a database sized to Java requirements and Java installed inside the database.

When you try to enable Java in an existing Oracle database, you can use the Java related parts of this procedure as an example and modify it according to your needs, that is, increase dbsize, increase `shared_pool_size`, create a large rollback segment, run `initjvm.sql`, and so on. For more information, refer to *Oracle Database Java Developer's Guide*.

Where can files related to Java reside and how should they be encoded?

It is not absolutely straightforward where files used by Java have to be stored and how they should be encoded. In general files can reside in native BS2000 or in the POSIX file system, but there are exceptions.

The following table gives an overview of the file types, location, default encoding, and encoding modifications for APIs or statements

Statement or API	File type	Place	Default encoding	Encoding modification
`CREATE JAVA CLASS USING BFILE`	.class	BS2000 PAM file or POSIX	Binary	Not applicable
`CREATE JAVA RESOURCE USING BFILE`	.properties	BS2000 PAM file or POSIX	ascii	None, that is, there is no means to change default encoding)

Statement or API	File type	Place	Default encoding	Encoding modification
`CREATE JAVA SOURCE USING BFILE`	.java, .sqlj	BS2000 PAM file or POSIX	DB charset	Execute `dbms_java. set_compiler_option (' ', 'encoding', 'ISO8859-1')`
`CREATE JAVA SOURCE AS`	.sql	Part of statement	Session character set specified in `NLS_LANG`	`NLS_LANG`
`CALL DBMS_JAVA.LOADJAVA`	*, .jar, .zip	POSIX	DB charset	Option encoding in loadjava call
`java.io-package`	*	POSIX	DB charset	Depends on the classes used

BS2000 PAM files in ascii can be created by transferring files (FTP) from an ascii platform to BS2000 in binary mode.

The distinction between a native BS2000 file name and a POSIX file name is made by the preceding slash ('/'). As a consequence, no relative path names are allowed for POSIX file names.

However, there is one exception: when used within `dbms_java.loadjava`, relative path names are preceded by the value of `ORACLE_HOME` in `ORAENV` file.

12.2 Database character sets and Java Encodings

As far as I/O is concerned, the Oracle JAVAVM uses the database character set as system property `file.encoding`. Therefore the following Oracle/BS2000 database character sets have been added to the list of supported Java encodings:

```
WE8BS2000
WE8BS2000E
EE8BS2000
CE8BS2000
CL8BS2000
WE8BS2000L5
```

These encodings are not known to any other Java implementation.

The system property `file.encoding`, however, does not apply to Java property files. Property files always use the encoding 8859_1 (refer to *Oracle Database SQLJ Developer's Guide*). The system property `file.encoding` is used when compiling a source file. You can change this default by either using the following procedure or by setting the encoding option of the procedure `dbms_java.loadjava`:

```
dbms_java.set_compiler_option('','encoding',...)
```

12.3 Java Demonstration Files

A simple Java demonstration program running in the server is shipped under:

```
$ORAC1120.JAVAVM.DEMO.HELLO.SQL
```

An example with database connection using the server-side internal driver is shipped under:

```
$ORAC1120.JAVAVM.DEMO.EMPLOYEE*
```

13

XML

This chapter describes BS2000 specific topics of XML such as installation, features, and restrictions. For more information refer to *Oracle XML Developer's Kit Programmer's Guide*, *Oracle XML DB Developer's Guide*, *Oracle Database XML C API Reference*, *Oracle Database XML C++ API Reference*, and *Oracle Database XML Java API Reference*.

13.1 Installation of Xdk

The Oracle XML Developer's Kit (Xdk) is not part of the Oracle Database Product Set. Oracle recommends downloading the latest version of Xdk from the Oracle Technical Network OTN (`http://www.oracle.com/technetwork/topics/index.html`). The Jar files for the XML SQL utility (`xsu12.jar` and `xdb.jar`) and the XML Parser (`xmlparserv2.jar`) are already loaded in the database when you have followed the procedures explained in Chapter 4, "Creating and Upgrading a Database".

13.2 Features and Restrictions

The following table provides an overview of which XML features are available for which languages on BS2000.

The meaning of empty fields is: not supported.

N/A means: not applicable.

	Availability for			
XML Feature	**Java**	**C**	**C++**	**PL/SQL**
Parser	Yes			Yes
XSLT Processor	Yes			Yes
Class Generator		N/A		N/A
XSQL		N/A	N/A	N/A
Transviewer Beans		N/A	N/A	N/A
XML-SQL Utility	Yes	N/A	N/A	Yes
Schema Processor	Yes			N/A

When using PL/SQL instead of Java you should consider the following behavior:

- PL/SQL File input is only possible from POSIX and with ASCII-Data Format.

- PL/SQL File output is written to POSIX with ASCII-Data Format.
- For `INSERT/UPDATE/DELETE` operations the XML document must not contain `<?xml ... encoding=WE8BS2000 ...>`.

When using the JAVA-interfaces you must ensure the right charset of the data. For more information, refer to the encoding considerations listed in Chapter 12, "Java in the Database".

If you have an ASCII platform with JDK, then you can also use XML components and operate on the BS2000 Oracle database using a JDBC connection.

14

Oracle Management Agent

This chapter describes the Oracle Management Agent for BS2000/OSD and provides installation and configuration information. The reader is supposed to be familiar with Oracle Enterprise Manager.

The Oracle Management Agent for BS2000/OSD is responsible for monitoring all components on the host computer. Once installed, the Oracle Management Agent knows how to monitor default target types, such as the Oracle Database. For more information, refer to *Oracle Enterprise Manager Concepts*.

This chapter describes the following sections:

- Preinstallation Issues
- Running the Installation Script
- Running the Agent
- Restrictions
- Troubleshooting

14.1 Preinstallation Issues

The following is a list of preinstallation issues:

- Check if the BS2000/OSD POSIX subsystem is started
- Check if the *open*Net Server tool `netstat` is installed under POSIX.
- Check if jenv v6.0 is installed
- Check if perl v5.8 is installed

  If one of the required software packages is not installed, then install the software package first before installing Oracle Management Agent.
- Ensure that the user address space is 1 GB
- Ensure that there is at least 1 GB disc space available on the POSIX file system
- Ensure that the file `/etc/hosts` includes an entry with a loopback address for the localhost similar to the following example:

  ```
  127.0.0.1 localhost local # loopback
  ```

- Add all Oracle databases you want to monitor the file `/var/opt/oracle/oratab` with the following format:

  *ORACLE_SID:ORACLE_HOME:*N|Y*:ORACLE_DATA*

The following table describes the parameters in the format:

Parameter	Description
ORACLE_SID	system identifier of the database
ORACLE_HOME	Oracle home directory under POSIX
ORACLE_DATA	user ID where the database files are stored

Each database that should be monitored needs an entry in the oratab file. An example for a valid oratab file is:

```
# oratab file
# valid entries are of the following form:
# ORACLE_SID:ORACLE_HOME:N|Y:ORACLE_DATA
orcl:/ora11202/oracle/product/10g:N:$ORACLE
```

Note: The DBA group should have read and write access to the oratab file and the oracle directory in the /var/opt folder.

- Check the Oracle Management Agent release number. Oracle Management Agent that ships with release 11.2 can monitor releases 10.x and 11.x. Monitoring of a Oracle9*i* is not supported.
- Ensure that you copy the appropriate LISTENER.ORA file to the $ORACLE_HOME/network/admin directory to monitor a LISTENER.
- Ensure that you grant access to the alert file to all users, if you want to monitor a database under a user ID different from the user ID of the Enterprise Management Agent.

14.2 Running the Installation Script

It is recommended to install the Oracle Management Agent under a separate Oracle Home directory. All Oracle users should belong to the same group, for example, oracle. To start the installation enter:

```
/CALL-PROCEDURE $ORACLE1120.INSTALL.P.AGENT
```

You are prompted for the following:

Parameter	Description
EMDROOT	This is the Management Agent's home directory.
EM-AGENT-PORT	The port on which the Management Agent should listen.

Parameter	Description
EM-OMS-HOST	The host name of the Oracle Management Server
EM-OMS-HTTP-PORT	The port of the Oracle Management Server

To finalize the EM Agent installation login as root and execute the following script:

```
$EMDROOT/root.sh
```

The installation of the Management Agent, can also be operated by a background process. In this case, the parameters EMDROOT, EM-AGENT-PORT, EM-OMS-HOST and EM-OMS-HTTP-PORT are mandatory parameters and must be specified with the following procedure call:

```
/ENTER-PROCEDURE
$ORAC1120.INSTALL.P.AGENT,(EMDROOT='/u01/app/orac1120/product/emagent',EM-AGENT-PO
RT='1813',EM-OMS-HOST='omshost.example.com',EM-OMS-HTTP-PORT='1159'),
CPU-LIMIT=300
```

Note:

- The installation is logged to the INSTALL.EMAGENT.LST file.
- It is recommended to use a CPU-LIMIT greater than 200.

14.3 Running the Agent

- To run the agent, start a POSIX shell and type the following command:

  ```
  emctl start agent
  ```

  If you start the Management Agent with the emctl utility in the POSIX shell, then the Management Agent inherits the JOB-CLASS, CPU-LIMIT and RUN-PRIORITY of the login process.As a consequence a defined CPU-LIMIT causes the Management Agent to terminate when the limit is reached. If you want to start the Management Agent without CPU-LIMIT, then use the BS2000 start procedure for the Management Agent. Login to the BS2000 Management Agent user ID and execute the command:

  ```
  /ENTER-PROCEDURE
  ST-AGENT.PRC,JOB-NAME=EMAGENT,JOB-CLASS=<ntl-j-c>,CPU-LIMIT=*NO,RUN-PRIO=210
  ```

- To stop the agent, start a POSIX shell and type the following command:

  ```
  emctl stop agent
  ```

- To get the status of the agent, start a POSIX shell and type the following command:

  ```
  emctl status agent
  ```

Use the Oracle Management Agent for BS2000/OSD as described in the *Oracle Database 11g Enterprise Manager* book set.

Note: Oracle recommends to use the start procedure in the BS2000 environment to start the Management Agent.

14.4 Restrictions

The following is a list of issues that might affect Oracle Enterprise Manager Agent

- Oracle Management Agent for BS2000/OSD can monitor servers that are running Oracle Database 10*g* or Oracle Database 11*g*.
- The Oracle Management Agent for BS2000/OSD does not support the SNMP framework. That means, the agent is not integrated within the EMANATE master agent for BS2000/OSD. The agent does not accept any SNMP requests and does not send any SNMP trap. Therefore, the Oracle Management Agent for BS2000/OSD is not usable with third party SNMP management systems.
- Oracle Database 10*g* and Oracle Database 11*g* for BS2000/OSD supports only a minimal set of system statistics and these may not reflect the actual performance of the system.
- Currently, Oracle Database 11*g* for BS2000/OSD only ships with the Oracle Enterprise Manager Agent. Oracle Enterprise Manager Grid Control Console is not supported, though this release does support a remote Oracle Enterprise Manager Grid Control Console.
- Currently, the Management Agent supports only a minimal set of host metrics and these may not reflect the actual performance and only a part of the configuration of the system.
- The dbconsol is not supported on BS2000/OSD.

14.5 Troubleshooting

Typically, the Oracle Management Agent is monitored by an internal watchdog process and is restarted in case of an error. This method does not prevent from any crashes where at least one task of the thread application is kept alive and might inhibit a proper restart of the Management Agent. If the Oracle Management Agent cannot be started or restarted, then use the following instructions to resolve the issue:

```
/CALL-PROCEDURE $TSOS.SYSPRC.PTHREADS.010(ITH-SHOW)
```

Login to the POSIX shell, set your environment and run the utility ithshow, for example:

```
$ . /u01/app/orac1120/agent11g/.profile.oracle
$ ithshow
```

This utility produces an output similar to the following:

```
STARTED AT 2012-06-12-135624 BY POSIX (running)
LLM = EMAGENT (prelinked)
MAIN = IC@#MAIN
APPLL = :POR2:$ORAQAX13.ORALOAD-ASCII.LIB
RUNTL = :BUG1:$TSOS.SYSLNK.PTHREADS.013
PTHvers = 01.3A10 2011-04-14 09:19:23
FDs = 8 (8 ORIG FDs)
Threads = 10 (6 user threads, 4 system threads)
TYPE TSN PID JOB-TYPE PRI CPU-USED CPU-MAX ACCOUNT#
ORIG 1MYQ 3942 (X'0F66') 3 DIALOG *0 240 16.7989 32767 FSC
executing request
THRE 1MYW 3948 (X'0F6C') 3 DIALOG *0 240 21.7415 32767 FSC
executing user thread
```

Here you find the TSNs of the tasks involved in the PTHREADS application. You can connect to the PTHREADS application when you choose the TSN of the ORIG task as the input for the parameter TSN of the ITH-START procedure in the following format:

```
/CALL-PROCEDURE $TSOS.SYSPRC.PTHREADS.013(ITH-START),(TSN=1MYQ)
```

When you see the double slash prompt you can type CANCEL-THREADED-PROGRAM to cancel the PTHREADS application. If the ORIG task is already terminated, then you can terminate all other PTHREADS tasks by using the BS2000 system command

```
/CANCEL_JOB JOB-IDENTIFICATION=tsn
```

If the Management Agent is running on a SQ system in x86-64 mode, then you must define the proper load library in the procedure parameters as follows:

```
/CALL-PROCEDURE
$TSOS.SYSPRC.PTHREADS.013(ITH-START),(START-LIBRARY=$TSOS.SKULNK.PTHREADS.013,TSN=
1MYQ)
```

If the Management Agent terminates abnormally, then it might be possible that a process with the name dblu*user_number* stays alive. This process is a program cache and cannot be canceled with the BS2000 CANCEL-JOB command. This process can only be stopped using the POSIX command posdbl. For example, the following shows you how you can identify and stop a pending program cache in the POSIX shell:

```
$ ps -fu emuser
UID PID PPID C STIME TTY TIME CMD
EMUSER 2178 2175 0 07:48:08 pts/0 0:01 [sh]
EMUSER 4179 2178 2 14:05:16 pts/0 0:00 [ps]
EMUSER 4168 1 0 14:05:06 ? 0:00 dblu511
```

In this example, the process dblu511 with the pid 4168 is still alive. To stop this process enter the following command:

```
$ pdbl -uD
```

15

Oracle on SQ Server

This chapter describes features, which are specific to Oracle Database for BS2000/OSD SQ Server:

- Concepts
- Definitions
- Product Set
- Hardware Requirements
- Software Requirements
- Address Space Limit
- Installing Oracle Database Software
- Database Creation
- Upgrade and Migration
- Administration
- Oracle application programs
- openUTM Product Set

15.1 Concepts

The SQ server line is a line of BS2000/OSD business servers, which are based on x86-64 architecture. Oracle Database 11*g* supports this innovative line of SQ servers, for example, SQ200, and the BS2000 operating system OSD/XC.

Oracle Database 11*g* Release 2 for SQ Server runs in native x86-64 mode in the BS2000 partition of the SQ Server. Programs running in this mode provide much better performance than programs running in /390 mode (= compatibility mode).

There are no differences in handling and administration compared to the "conventional" Oracle for BS2000 (on /390 systems). So the migration from conventional Oracle for BS2000 to Oracle for SQ Server is very simple.

All products you know from the /390 edition of Oracle, for example, Oracle Net Services, SQL*Plus, Precompilers, are available in native x86-64 mode for the SQ Server line.

15.2 Definitions

Because there are two Oracle editions for BS2000, that is, one for the /390 series and one for the SQ series, the following definitions are introduced to avoid confusion:

Oracle edition	Meaning
Oracle for BS2000/390:	BS2000 line of Oracle, which is running on systems with /390 CPU's (S Servers). This Oracle line can also run on SQ Server, but only in the slower /390 mode, also called compatibility mode. Synonyms: "/390 edition of Oracle/BS2000"
Oracle for BS2000/x86-64:	BS2000 line of Oracle, which runs in native x86-64 mode in the BS2000 partition of the SQ Server. This Oracle line cannot run in the Linux or Windows partition of the SQ Server. This Oracle line cannot run on /390 CPU's. Synonyms: "SQ edition of Oracle/BS2000" "Oracle for SQ series" "Oracle for Fujitsu BS2000/OSD (SQ series)" "Oracle for SQ Server line"

15.3 Product Set

All products, which are available with the /390 edition of Oracle/BS2000 are also available with the x86-64 edition. There are no differences in functionality between the products of these two Oracle/BS2000 editions.

15.4 Hardware Requirements

- SQ Server, for example SQ200
- Main memory: at least 1024 MB
- Disk Space: Total static Oracle Database non-database requirements are approximately 2500 MB in the BS2000 file system and a minimum of 400 MB in the POSIX file system.

  The staging library for the installation requires about 2500 MB in the BS2000 file system.
- Database Space: No difference to /390 edition of Oracle

15.5 Software Requirements

The following table lists the software requirements:

Component	Requirement
Operating System	OSD/XC V4.1 or higher The BS2000/OSD subsystem POSIX should have been started.

Component	Requirement
Compilers	Though not required to run the Oracle Database, if high-level languages, such as C or COBOL, are used to interface to the Oracle Database, then the following compilers are supported: COBOL85 V2.3 CCOBOL2000 V1.5 and higher CPP as of V3.1
openUTM	openUTM V6.0 and higher
CRTE	CRTE V2.8 and higher

15.6 Address Space Limit

A user's address space should not be less than 512 MB. Oracle recommends loading the x86-64 version of BS2000 DSSM "CRTE-BASYS" (= CRTEBASK) to reduce memory requirements.

15.7 Installing Oracle Database Software

Refer to Chapter 3, "Oracle Database Installation and Deinstallation".

15.8 Database Creation

The creation of a database works exactly as described in Chapter 4, "Creating and Upgrading a Database". You may create a database automatically, by BS2000 procedure `INSTALL.P.SUPER`, or manually.

There are no differences to the /390 edition of Oracle for BS2000.

15.9 Upgrade and Migration

Refer to Chapter 4, "Creating and Upgrading a Database".

15.10 Administration

There are almost no differences in administration compared to the /390 edition of Oracle Database. The only difference is that Oracle on SQ server does not support executables ("phases"), but only LLMs. Therefore, you must start Oracle programs like SQLPLUS as follows:

```
/START-PROGRAM *MODULE($ORAC1120.ORALOAD.LIB,SQLPLUS,A,A)
```

15.11 Oracle application programs

Oracle for SQ Server supports the same programming languages for Oracle application programs as the /390 edition of Oracle. There are Precompilers for C/C++ and Cobol.

Oracle application programs can run on SQ Servers only in /390 mode.

Application programs have to be linked as described in the *Oracle Database User's Guide for Fujitsu BS2000/OSD*.

Application programs, which have been developed on /390 systems can run on SQ Servers without any changes.

15.12 openUTM Product Set

The openUTM product set of Oracle Database 11*g* Release 2 (11.2) for SQ Server requires openUTM V6.0 and higher.

There are no differences in developing Oracle Database/openUTM applications in comparison with the /390 edition of Oracle (refer to chapter "Using the Oracle Database under openUTM" in the *Oracle Database User's Guide for Fujitsu BS2000/OSD*).

A

Oracle Error Messages for BS2000/OSD

This appendix lists Oracle Database 11*g* Release 2 for Fujitsu BS2000/OSD messages with possible causes and suggested actions. The messages shown in this chapter may be accompanied by additional text when displayed on screen. This text identifies the function that detected the problem, and can include internal status codes, BS2000 system macro return codes, or both. These codes can be helpful to the Oracle Support Services Representative in determining the cause of a problem.

Sometimes, for example, in the early stages of initialization when the message components are not yet available, the Oracle Database cannot issue a regular Oracle message. If this occurs, then Oracle Database calls the ILCS task termination routine, or it issues a `TERM` macro directly, giving the message number as the user termination code. You can use this message number to find the explanation in this appendix.

ORA-05000: ORACLE termination routine called

Cause: The termination routine of the Oracle Database run-time system has been called due to a fatal error.

Action: If you do not know why the Oracle Database program terminated, or how to resolve this problem, then contact the Oracle Support Services Representative.

ORA-05001: Unsupported BS2000 Version

Cause: The active version of the BS2000/OSD operating system is not supported by Oracle Database 11*g* Release 2.

Action: Upgrade to a more recent BS2000/OSD version.

ORA-05002: Fatal error: called from non-ILCS program

Cause: In a precompiler or OCI application, the Oracle Database is called from a program that does not run in an ILCS environment. The Oracle Database does not support non-ILCS programs

Action: Ensure that the application program runs in ILCS mode. Some programming languages, for example, FOR1, PL/I, require specific options for ILCS. Refer to the Fujitsu documentation for further information.

ORA-05003: Fatal error: ILCS PCD cannot be verified

Cause: In a precompiler or OCI application, Oracle Database is called with a save area that is marked as an ILCS save area but does not point to a proper PCD (ILCS global area). The problem is either that memory has been overwritten, or that Oracle Database is called from a program that does not run in an ILCS environment. Oracle Database does not support non-ILCS programs.

Action: Ensure that the application program runs in ILCS mode. Some programming languages, for example, FOR1, PL/I, require specific options for ILCS. Refer to the Fujitsu documentation for further information.

ORA-05004: Fatal error: stack overflow, extension failed

Cause: A call to a function required an extension of the current call stack segment. This extension failed and the corresponding ILCS routine returned the error.

Action: Ensure that the user address space is large enough in the JOIN entry and that there is no temporary memory saturation. Then re-run the program. If you need further help, then contact the Oracle Support Services Representative.

ORA-05005: Error: IT0INITS called in PROLOD

Cause: This is an internal error and should not occur.

Action: Contact the Oracle Support Services Representative.

ORA-05006: sltga already initialized

Cause: The initialization routine for the sltga is called more than one time.

Action: Check if more than one stub modules (`PROSTUB`, `XAOSTUB`) are linked to the application.

ORA-05007: failed to load OSNTAB

Cause: This message is usually be preceded by a BS2000 BLS-*nnnn* message. The most likely reason is that the `ORALOAD` library cannot be found.

Action: Contact the Database Administrator about the ORALOAD library. If you cannot identify the cause of the problem, then contact the Oracle Support Services Representative.

ORA-05008: failed to load requested network driver

Cause: This message is usually be preceded by a BS2000 `BLS-nnnn` message. The most likely reason is that the `ORALOAD` library cannot be found.

Action: Contact the Database Administrator about the `ORALOAD` library. If you cannot identify the cause of the problem, then contact the Oracle Support Services Representative.

ORA-05009: osnsgl: user connects invalid in kernel

Cause: A database link was set up using the single-task driver (S:). This is invalid, as the single-task driver can only be used for call connections on the user side.

Action: Select a different network driver for the database link.

ORA-05010: bad filename length

Cause: Buffer overflow while building/translating a file name. This could be caused by specifying an excessively long file name in the `ORAENV` file.

Action: If you cannot identify the cause of the problem, then contact the Oracle Support Services Representative.

ORA-05011: bad file size

Cause: This is an internal error and should not normally occur.

Action: Contact the Oracle Support Services Representative.

ORA-05012: bad block size

Cause: This is an internal error and should not normally occur.

Action: Contact the Oracle Support Services Representative.

ORA-05013: bad filename parse

Cause: A file name being analyzed is not well-formed for Oracle Database purposes.

Action: Correct the file name and re-run the program.

ORA-05014: sfcopy: non-matching block size

Cause: In a partial database file copy, source and target file have different block sizes. This may indicate an internal error and should not normally occur.

Action: If you cannot identify the cause of the problem, then contact the Oracle Support Services Representative.

ORA-05015: text file open failed

Cause: An Oracle Database text or command file cannot be opened. One of the following could cause this error: the file name is wrong, the file has not been properly initialized, or the file is not accessible.

Action: Correct the problem and restart the Oracle Database. If this occurs when you issued the `STARTUP` command, then check the initialization file for the correct specification of the database files.

ORA-05016: text file close failed

Cause: Attempt to close an Oracle Database file has failed. This is an internal error and should not normally occur.

Action: Contact your Oracle Support Services Representative.

ORA-05017: file open failed

Cause: An Oracle Database database file cannot be opened. Either the file name is wrong, the file has not been properly initialized, or the file is not accessible (for example a file may not be accessible for a cross-user ID single-task client).

Action: Correct the problem and restart the Oracle Database. If this occurred when you issued the `STARTUP` command, then check the initialization file for the correct specification of the database files.

ORA-05018: file seek failed

Cause: This is an internal error and should not normally occur.

Action: Contact the Oracle Support Services Representative.

ORA-05019: file write failed

Cause: An I/O error occurred while writing to an Oracle Database file.

Action: If the error cannot be identified as one caused by a disk malfunction, then either contact the System Administrator, or contact the Oracle Support Services Representative.

ORA-05020: write block outside of file

Cause: An attempt was made to write a block of an Oracle Database file that does not exist. For example, block number < `1` or > file size. This is an internal error and should not normally occur.

Action: Contact the Oracle Support Services Representative.

ORA-05021: file read failed

Cause: An I/O occurred while reading an Oracle Database file.

Action: If the error cannot be identified as one caused by a disk malfunction, then either contact the System Administrator, or contact the Oracle Support Services Representative.

ORA-05022: read block outside of file

Cause: An attempt was made to read a block of an Oracle Database file that does not exist. For example, block number < 1 or > file size. This is an internal error and should not normally occur.

Action: Contact the Oracle Support Services Representative.

ORA-05023: file close failed

Cause: The attempt to close an Oracle Database file failed. This is an internal error and should not normally occur.

Action: Contact the Oracle Support Services Representative.

ORA-05025: sfccf:file mismatch. Trying to reuse a file with different size

Cause: When trying to reuse a database file, the file size specified differs from the actual size of the existing file.

Action: Specify the correct file size (remember to subtract one logical block for the implicit header block), or leave the size unspecified, or use a different file name if you want to create a larger or smaller database file.

ORA-05026: file does not exist

Cause: An attempt was made to access a database file, which no longer exists.

Action: Contact the Database Administrator who may know why this error has occurred. If the Database Administrator cannot find the cause of the problem, then contact the Oracle Support Services Representative.

ORA-05027: file does exist

Cause: When attempting to create a new file, this error occurs if the file is found and is not empty.

Action: If the error occurred in a `create database`, then retry with the `reuse` option. Otherwise use a different file name or check whether the file can be erased.

ORA-05028: file is not a dbfile

Cause: The database, or log, or control, file to be opened does not contain the proper identification for such a file.

Action: Check for wrong file specification.

ORA-05029: illegal use-option

Cause: Internal error. Function `sfccf` was called with an illegal option.

Action: Contact the Oracle Support Services Representative.

ORA-05030: SID not defined

Cause: When the system id was required, typically, to substitute the "?" in names, for example, in file names set by the initialization file, it was not yet defined. This could be caused by a missing `ORAENV` file or a missing `ORASID` in that file.

Action: Ensure that the `ORAENV` file definition is correct and re-run the program.

ORA-05031: SID translation failure

Cause: The system id is syntactically incorrect.

Action: Ensure that the `ORASID` definition is correct and re-run the program.

ORA-05032: bad name parse

Cause: The translation of a file name, or other name containing variable parts, failed. The error may be caused by a wrong specification in the `ORAENV` file.

Action: Ensure that the `ORAENV` variable assignments are correct. If you cannot identify the cause of the problem, then contact the Oracle Support Services Representative.

ORA-05033: bad environment values

Cause: One or more of the values specified in the `ORAENV` file are invalid.

Action: Ensure that you specified legal values in the `ORAENV` file, refer to the Appendix B, "Oracle Environment Variables" in this guide or *Oracle Database User's Guide for Fujitsu BS2000/OSD* for further information.

ORA-05034: bad seal

Cause: Internal error. An internal file control structure is found to be corrupt.

Action: Contact the Oracle Support Services Representative.

ORA-05035: host command not executed

Cause: A BS2000 command, argument of a `HOST` or `#HOST` command, is invalid or too long.

Action: Enter a valid `HOST` command.

ORA-05036: bad user id (*length*)

Cause: Internal buffer overflow while building a file name from variable components.

Action: Ensure that the `ORAUID` value specified in the `ORAENV` file is correct. If you cannot identify the cause of the problem, then contact the Oracle Support Services Representative.

ORA-05037: /CANCEL command not executed

Cause: A background job could not be cancelled. The background task may have already been terminated.

Action: If you cannot identify the cause of the problem, then contact the Oracle Support Services Representative.

ORA-05038: SID has illegal length

Cause: The system identifier specified in either the `ORAENV` file or as part of a connect string exceeds 4 characters in length.

Action: Specify a correct value.

ORA-05039: Recursive entry to ssodrv

Cause: Oracle Database kernel has been reentered at the top. This should not happen.

Action: Ensure that the user program does not incorrectly call Oracle Database functions from within an interrupt handling routine (signal routine, contingency). If you cannot identify the cause of the problem, then contact the Oracle Support Services Representative.

ORA-05040: no more dynamic memory

Cause: Request memory failed in file-management components. This is probably caused by a user address space that is too small.

Action: Ensure that the user address space is large enough in the `JOIN` entry and that there is no temporary memory saturation. Then re-run the program. If you need further help, then contact the Oracle Support Services Representative.

ORA-05041: Interrupt in soarch

Cause: The archiver process was unexpectedly interrupted.

Action: If you cannot identify the cause of the problem, then contact the Oracle Support Services Representative.

ORA-05042: soarch: Buffer overflow

Cause: The archiver process detected an internal buffer overflow.

Action: If you cannot identify the cause of the problem, then contact the Oracle Support Services Representative.

ORA-05043: Archive control string too long

Cause: The archive control string is too long.

Action: Shorten this parameter and restart the database.

ORA-05044: Archive generated filename too long

Cause: The file name is generated from the values of the initialization parameters `log_archive_format` and `log_archive_dest`. This has resulted in a file name that is too long.

Action: Run the command `ALTER SYSTEM ARCHIVE LOG START TO VALID_DEST` where `VALID_DEST` is a valid BS2000 file name.

ORA-05045: Archive file creation/open error

Cause: The archive file is normally allocated dynamically. Either this or the subsequent open failed. Possible causes are either insufficient space left on disk, or a bad archive file allocation parameter in `ORAENV`.

Action: Ensure that the optional `ORAENV` parameter is correct and that sufficient disk space is available.

ORA-05046: Archive control string error

Cause: The archive file name or control parameters are incorrect.

Action: Correct the parameters.

ORA-05050: PGA (fixed part) could not be allocated

Cause: Probable operating system error or internal error.

Action: Contact the Oracle Support Services Representative.

ORA-05051: cannot allocate var. PGA

Cause: During creation of the PGA, required dynamic memory could not be allocated.

Action: Verify that the user address space is large enough and that if an application program produced the error, the program is not consuming excessive memory. Otherwise contact the Oracle Support Services Representative.

ORA-05052: error deleting var. PGA

Cause: During deletion of the PGA, dynamic memory could not be released. This is an internal error and should not normally occur.

Action: Contact the Oracle Support Services Representative.

ORA-05053: invalid or missing PGA_BASE

Cause: An invalid value for the `PGA_BASE` parameter has been specified in the `DBA ORAENV` file.

Action: Use the default value for the `PGA_SIZE` environment variable. If this does not solve the problem, then contact the Oracle Support Services Representative.

ORA-05054: invalid or missing PGA_SIZE

Cause: An invalid value for the PGA_SIZE environment variable has been specified in the DBA ORAENV file. You should never need to change the default value for the PGA_SIZE environment variable.

Action: Use the default value for the PGA_SIZE environment variable. If this does not solve the problem, then contact the Oracle Support Services Representative.

ORA-05055: address range for PGA (fixed part) is not free

Cause: The address range described by the PGA_BASE and PGA_SIZE ORAENV variables is not available for allocation. This may be due to overlapping PGA, SGA, and KERNEL areas, or to an application program, which has occupied memory in this area. If you did not specify a value for PGA_BASE, the default may be inappropriate for the case.

Action: Refer to the section "Address Space Planning" in Chapter 2.4, "Address Space Planning" for further information.

ORA-05056: no more context space

Cause: During processing of a SQL request, dynamic memory could not be allocated. This could happen when very complex requests are being processed and there is not enough memory available.

Action: Verify that the user address space is large enough and that the application program, if the error occurred when you were using an application program, is not using excessive memory. Otherwise, contact the Oracle Support Services Representative.

ORA-05058: assert failed: SGA not mapped

Cause: This is an internal error and should not normally occur.

Action: Contact the Oracle Support Services Representative.

ORA-05059: assert failed: not in kernel

Cause: This is an internal error and should not normally occur.

Action: Contact the Oracle Support Services Representative.

ORA-05060: SGA not created

Cause: After you run the STARTUP command, the SGA shared memory pool could not be created.

Action: Verify that you are not trying to start the database while it is running and that the database system id is not being used for two different databases. Otherwise, contact the Oracle Support Services Representative.

ORA-05061: SGA attach failed

Cause: Connection to the SGA shared memory pool could not be established. This may have happened if you used the wrong system id, or if the database you expected to be running is not running.

Action: Verify that it is not one of the preceding causes (check with the Database Administrator). Otherwise, contact the Oracle Support Services Representative.

ORA-05063: SGA base invalid

Cause: An invalid value has been specified for the SGA_BASE parameter in the ORAENV file.

Action: This value is not normally needed. If specified, it must be a hexadecimal value giving the full virtual address for the SGA memory pool. Correct the value and run the `STARTUP` command.

ORA-05064: cannot allocate SGA

Cause: After creating the memory pool, the `REQMP` to allocate the space failed. This might be an operating system error.

Action: If you cannot identify the cause of the problem, then contact the Oracle Support Services Representative.

ORA-05065: SGA not deleted

Cause: When attempting to detach from the SGA, the `DISMP` system macro returned an error.

Action: If you cannot identify the cause of the problem, then contact the Oracle Support Services Representative.

ORA-05066: SGA address space conflict

Cause: The SGA cannot be placed at the requested address range, because the range is already partly used. The SGA start address is defined by the `ORAENV` variable, `SGA_BASE`; its size is determined by various initialization file parameters such as processes, buffers, and so on.

Action: Refer to the section on "Address Space Planning" in Chapter 2, "Architecture and Implementation", and adjust the relevant initialization file and `ORAENV` variables. Inspect the `JOIN` entry for the address space limit. Contact the System Administrator to find out about shared subsystems and their placement in the address space. Ensure that you do not overlap with the Oracle Database kernel.

ORA-05067: SGA: address space saturation

Cause: When the SGA is being allocated, the operating system reported that the virtual address space is saturated.

Action: Contact the System Administrator about paging area size and current overall system load.

ORA-05068 SGA still active, should not be

Cause: When the SGA is being created during startup, it is found that the SGA memory pool is still in use, although the databases should be shut down. This may be caused by a hanging single-task, user task, or a network server task.

Action: Check for such hanging tasks. Cancel these tasks, and then restart the database.

ORA-05069: Unexpected SGA memory pool problem

Cause: The `ENAMP` macro returned an unexpected error code.

Action: Contact the Oracle Support Services Representative.

ORA-05070: cannot enable TPA ser.item

Cause: Probable operating system error.

Action: Contact the Oracle Support Services Representative.

ORA-05071: cannot ENQ on TPA ser.item

Cause: Probable operating system error.

Action: Contact the Oracle Support Services Representative.

ORA-05072: cannot enable post/wait item

Cause: Probable operating system error.

Action: Contact the Oracle Support Services Representative.

ORA-05073: error in post

Cause: An inter-process communication operation failed.

Action: Check that the database and all required background tasks are running correctly. If you cannot identify the cause of the problem, then contact the Oracle Support Services Representative.

ORA-05074: error in wait

Cause: An inter-process communication operation failed.

Action: Check that the database and all required background tasks are running correctly. If you cannot identify the cause of the problem, then contact the Oracle Support Services Representative.

ORA-05075: error in task table manager

Cause: Internal error.

Action: Contact the Oracle Support Services Representative.

ORA-05076: error setting spid

Cause: Probable operating system error.

Action: Contact the Oracle Support Services Representative.

ORA-05077: cannot enable HIA event

Cause: Probable operating system error. The HIA (Here I Am) event item is used during startup to communicate between a started background task and the invoking SQL*DBA program.

Action: Contact the Oracle Support Services Representative.

ORA-05078: create process failure

Cause: When you issued the `STARTUP` command, a background job could not be started successfully.

Action: Check for any job scheduling problems and that any `BGJPAR` entry in the `ORAENV` file is correct. If you cannot identify the cause of the problem, then contact the Oracle Support Services Representative.

ORA-05079: internal asynchronous IO error

Cause: This is an internal error and should not normally occur.

Action: Contact the Oracle Support Services Representative.

ORA-05101: bind-error *xxxxxxxx* for module/library

Cause: The Oracle Database/UTM attach module could not be loaded. One possible reason is that the Oracle Database has been installed under a user ID different from `$ORAC1120` and that the installation procedure has not executed correctly.

Action: Ensure that the `ORAUID` definition in the `ORAENV` file is correct. Otherwise, contact the Oracle Support Services Representative.

ORA-05102: module verification failure: ORADBCN@

Cause: The openUTM application has probably been link-edited with an Oracle Database version different from the Oracle Database version used at execution.

Action: Re-link the UTM application. If the error persists, then contact the Oracle Support Services Representative.

ORA-05103: generated TSKM too short

Cause: The TSKM area was overwritten by the Oracle Database.

Action: Change the parameter `LTHTSKM` in the `KDCDB / KDCDBO` macro.

ORA-05104: generated TAM too short

Cause: The TAM area was overwritten by the Oracle Database.

Action: Change the parameter `LTHTAM` in the `KDCDB / KDCDBO` macro.

ORA-05107: POSIX environment variable <variablename> not defined

Cause: The specified environment variable is not defined.

Action: Define and export the requested variable in your profile.

ORA-05108: failed to process BS2000 command <bs2-command>

Cause: The BS2000 command processor cannot execute the command.

Action: Test the logged command in the POSIX shell using the POSIX command, `bs2cmd`.

ORA-05109: failed to initialize environment for POSIX

Cause: An application running under the POSIX shell cannot create links to required files in the BS2000 file system.

Action: Check if the environment Variables required for Oracle applications under POSIX are set properly.

ORA-05110: cannot attach to memory pool

Cause: Invalid pool ID parameter `xxx_MPID` or operating system error.

Action: Check the `ORAENV` parameter `xxx_MPID`, at most 4 characters of the set [A...Z],[0...9]), or contact the Oracle Support Services Representative.

ORA-05111: error attaching to memory pool

Cause: This is an internal error and should not normally occur.

Action: Contact the Oracle Support Services Representative.

ORA-05112: error creating memory pool

Cause: This is an internal error and should not normally occur.

Action: Contact the Oracle Support Services Representative.

ORA-05114: bad pool base

Cause: An invalid value for the base address parameter of the shared pool, that is, `COM_BASE`, and so on. has been specified in the `ORAENV` file.

Action: If this value is specified, it must be a hexadecimal value giving the full virtual address for the base address of a memory pool. Correct the value and restart the database.

ORA-05116: cannot load shared code into pool

Cause: Shared code could not be loaded into the specified memory pool. Generally, this message is preceded by a `BLS-nnnn` message from the operating system.

Action: Ensure that the ORALOAD link name identifies the correct ORALOAD library. Then restart the program. If you cannot identify the cause of the problem, then contact the Oracle Support Services Representative.

ORA-05117: cannot attach to socket subsystem

Cause: An application could not be bound to the sockets subsystem. Generally this message is preceded by a BLS-nnnn message from the operating system.

Action: Contact the Oracle Support Services Representative.

ORA-05118: ORACLE PCD slot not accessible

Cause: The current task is trying to attach to the ORACLE PCD slot but cannot find this slot.

This is an internal error and should not normally occur.

Action: Contact the Oracle Support Services Representative.

ORA-05119: module verification failed

Cause: The version of the shared loaded module does not match the version of the connection module on the user side.

Action: Contact the Oracle Support Services Representative.

ORA-05120: waiting for shared module to be loaded timed out

Cause: This is an internal error and should not normally occur.

Action: Contact the Oracle Support Services Representative.

ORA-05121: waiting for initialization of shared module timed out

Cause: This is an internal error and should not normally occur.

Action: Contact the Oracle Support Services Representative.

ORA-05126: Missing IT0PCD address

Cause: The ILCS run-time link-library is probably missing.

Action: Contact the System Administrator.

ORA-05127: PARAM-LIST AT CALL ORACLE NOT OK

Cause: System error.

Action: Contact the Oracle Support Services Representative.

ORA-05128: COMMIT/ROLLBACK/CONNECT NOT ALLOWED IN UTM-PROGRAM

Cause: Illegal SQL COMMIT/ROLLBACK/CONNECT found in UTM program.

Action: Correct the UTM program accordingly.

ORA-05131: ORADBCO-Call not allowed

Cause: System error.

Action: Contact the Oracle Support Services Representative.

ORA-05132: TA for User x is committed by the Resource-Manager

Cause: Transaction has been committed before malfunction.

Action: None.

ORA-05133: No Connect-String in Startparams found

Cause: In the start parameters there must be at least one open string for the Oracle Database.

Action: Refer to Chapter 8, "openUTM Product Set" of this guide and correct the start parameters.

ORA-05134: DBSTAT secondary opcode inconsistent

Cause: System error.

Action: Contact the Oracle Support Services Representative.

ORA-05135: Error x Recover PTC-list, Instance y

Cause: System error.

Action: Check if UTM is correctly installed and select privileges are granted to XA-tables, refer to Chapter 8, "openUTM Product Set", or contact the Oracle Support Services Representative.

ORA-05136: Maximum number of instances exceeded

Cause: The maximum number of open strings in the start parameters has been exceeded.

Action: Refer to Chapter 8, "openUTM Product Set" and correct the start parameters.

ORA-05137: Error x at Open Instance y

Cause: The connection to instance *y* is not possible.

Action: Start the Instance with all the required servers. If the error persists, then contact the Oracle Support Services Representative.

ORA-05138: Error x at Close Instance y

Cause: Error occurred when disconnecting from instance *y*.

Action: Because the disconnection has just been done, there is no action.

ORA-05139: Error x at Start Transaction for User z, Instance y

Cause: The start of transaction in instance *y* is invalid.

Action: Restart the UTM transaction. If the error persists, close the UTM application and do a restart.

ORA-05140: Error x at Continue Transaction for User z, Instance y

Cause: The continuation of transaction in instance *y* is invalid.

Action: Restart the UTM transaction. If the error persists, and the error is not 'XAER_PROTO'. This means that the transaction is rolled back because of longlock. Shut down the openUTM application and restart.

ORA-05141: Error x at Break Transaction for User z, Instance y

Cause: The break of transaction in instance *y* is invalid.

Action: Restart the UTM transaction. If the error persists, and there are no cursor operations in the preceded dialog step, shut down the UTM application and restart.

ORA-05142: Error x at End Transaction for User z, Instance y

Cause: The end of a transaction in instance y is invalid.

Action: Restart the UTM transaction. If the error persists, shut down the UTM application and do a restart.

ORA-05143: Mismatch in TA for User(s) z

Cause: In the UTM warmstart there is a mismatch between openUTM, and the Oracle Database instance(s), or both.

Action: Clear the Oracle Database instance(s), create a new `KDCDEF` for openUTM and restart the openUTM application.

ORA-05144: Error x at Prepare Transaction for User z, Instance y

Cause: The preparation for committing a transaction in instance y is invalid.

Action: Restart the UTM transaction. If the error persists, then shutdown the openUTM application and do a restart.

ORA-05145: Error x at Commit Transaction for User z, Instance y

Cause: Attempt to commit transaction in instance y unsuccessful.

Action: Restart the UTM transaction. If the error persists, then shutdown the openUTM application and restart.

ORA-05146: Error x at Rollback Transaction for User z, Instance y

Cause: The rollback of transaction in instance y is invalid.

Action: No action, but if the error persists, shutdown the UTM application and restart.

ORA-05147: TA for User z committed; Reason: Recovery

Cause: In a openUTM warmstart an interrupted transaction has been committed.

Action: None.

ORA-05148: TA for User z heuristic rolled back in Instance y

Cause: In a UTM warmstart, an interrupted transaction has just been rolled back from the Oracle Database.

Action: Restart the UTM transaction.

ORA-05149: TA for User z rolled back; Reason: Internal Event

Cause: The end- or prepare-call was invalid. Therefore the transaction must be rolled back.

Action: Restart the UTM transaction.

ORA-05150: KDCS-PEND before DBFITA missing

Cause: System error.

Action: Contact the Oracle Support Services Representative.

ORA-05151: KDCS-PEND before DBPETA missing

Cause: System error.

Action: Contact the Oracle Support Services Representative.

ORA-05152: Linked Resource-Manager is not CAE-compatible

Cause: System error.

Action: Contact the Oracle Support Services Representative.

ORA-05153: xa_switch definition not found for specified Resource-Manager: s

Cause: System error.

Action: Contact the Oracle Support Services Representative.

ORA-05154: Syntax error in start parameters for Resource-Manager: s

Cause: In the start parameters for the Oracle Database there is a syntax error.

Action: Refer to Chapter 8, "openUTM Product Set"and correct the start parameters.

ORA-05155: Internal Error: malloc in dbstpa

Cause: This error message indicates a system error. One of the possible reasons for this issue is that the system is out of memory.

Action: Contact the Oracle Support Services Representative.

ORA-05156: Internal Error: realloc in dbstpa

Cause: This error message indicates a system error. One of the possible reasons for this issue is that the system does not have sufficient memory.

Action: Contact the Oracle Support Services Representative.

ORA-05157: Internal Error: malloc in up_recovery

Cause: This error message indicates a system error. One of the possible reasons for this issue is that the system does not have sufficient memory.

Action: Contact the Oracle Support Services Representative.

ORA-05158: IUTMDB-Function not supported

Cause: openUTM system error.

Action: Contact the Oracle Support Services Representative.

ORA-05159: TA for User(s) z rolled back; Reason: Recovery

Cause: In a openUTM warmstart one or more interrupted transactions have been rolled back.

Action: None.

ORA-05161: TCP/IP can't perform asynchronous test on break socket.

Cause: Select on break socket failed.

Action: Contact the System Administrator about TCP/IP networking problems. If the error persists, contact the Oracle Support Services Representative.

ORA-05165: function not supported

Cause: Either Oracle Database 11*g* Release 2 Server or BS2000/OSD does not support this function.

Action: None.

ORA-05167: Defect in data buffer

Cause: This is an internal error and should not normally occur.

Action: Contact the Oracle Support Services Representative.

ORA-05170: SID not defined (ORAENV file missing?)

Cause: The system identifier, data base name, is not defined when needed during Oracle Database program initialization. A missing `ORAENV` file or a missing `ORASID` entry in that file could cause this error.

Action: Ensure that the `ORAENV` file definition is correct and re-run the program.

ORA-05173: bad kernel size

Cause: An invalid value for the `KNL_SIZE` parameter has been specified in the `ORAENV` file.

Action: You should not normally need to specify this variable, as the default value is correct. Contact the Oracle Support Services Representative.

ORA-05174: bad kernel base

Cause: An invalid value for the KNL_BASE parameter has been specified in the ORAENV file.

Action: If this value is specified, then it must be a hexadecimal value giving the full virtual address for the kernel memory pool. Correct the value and restart the database.

ORA-05175: Kernel address space conflict

Cause: The Oracle Database kernel cannot be placed at the requested address range, because the range is already used. The kernel start address is defined by the ORAENV parameter, KNL_BASE.

Action: Refer to the section on "Address Space Planning" in Chapter 2, "Architecture and Implementation", and adjust the relevant initialization file and ORAENV parameters. Inspect the JOIN entry for the address space limit. Contact the System Administrator to learn about shared subsystems and their placement in the address space.

ORA-05176: Kernel: address space saturation

Cause: When the Oracle Database kernel memory pool was being allocated, the operating system signalled that the virtual address space is currently saturated.

Action: Contact the System Administrator about paging area size and current overall system load.

ORA-05177: Unexpected Kernel memory pool problem

Cause: The ENAMP macro returned an unexpected error code.

Action: This problem can be caused when you run a program in 24-bit mode and try to connect single-task to a kernel, which has a value greater than the 16MB line, because the database itself is running in 31-bit mode. If this is the cause of the error, then you must access the database in two-task mode (through SQL*Net). Refer to the ENAMP macro description in the BS2000 documentation for other possible reasons. If you cannot identify the cause of the problem, then contact the Oracle Support Services Representative.

ORA-05178: Kernel module not yet initialized

Cause: The current task is trying to attach to an Oracle Database kernel which is not yet completely initialized. This can only happen if you try to connect to a database, which is just being started.

Action: Retry after a while. Remember that it may take a few minutes until a database is fully running and ready for the users. If the error persists, then check this issue with the Database Administrator.

ORA-05180: Cannot load character set table

Cause: One of the modules containing character set tables cannot be loaded.

Action: Verify that the ORALOAD library is accessible through LINK=ORALOAD. For further information about link names and the ORALOAD library, refer to the chapter "Getting Started" in *Oracle Database User's Guide for Fujitsu BS2000/OSD*. If you cannot identify the cause of the problem, then contact the Oracle Support Services Representative.

ORA-05181: load/init problem with PRO/OCI interface

Cause: The user-side stub module could not load the PRO/OCI module. In this case, the message is usually preceded by a BS2000 BLS-*nnnn* message, or the loaded module is incompatible with the version of the stub module.

Action: Ensure that the `ORALOAD` link name exists and points to the current `ORALOAD` library. Re-link the application with the current link libraries.

ORA-05191: symbol translation error for kernel memory pool

Cause: The logical name translation for the kernel memory pool failed. Normally, this indicates an invalid system id, `ORASID` in the `ORAENV` file.

Action: Ensure that the `ORAENV` file definition is correct. Otherwise, contact the Oracle Support Services Representative.

ORA-05192: cannot create/attach kernel memory pool

Cause: The memory pool for the Oracle Database kernel code could not be enabled. In a user program, a possible cause is that the user program already allocates part of the address range needed for the memory pool.

Action: Ensure that the user program does not request storage excessively, and that any `SGA_BASE` and `KNL_BASE` parameters in the `ORAENV` file are consistent. If you cannot identify the cause of the problem, then contact the Oracle Support Services Representative.

ORA-05193: Symbol translation error for kernel module or load library

Cause: The logical-name translation for the kernel module or load library failed. This is an internal error and should not normally occur.

Action: Contact the Oracle Support Services Representative.

ORA-05194: cannot load kernel

Cause: The kernel could not be loaded into the kernel memory pool. In most cases, this message is preceded by a `BLS-`*`nnn`* message from the operating system.

Action: Ensure that the `ORALOAD` link name identifies the correct `ORALOAD` library, and that the `ORAENV` variable, `KNL_MODULE`, names one of the possible kernels. Then re-issue the `STARTUP` command. If you cannot identify the cause of the problem, then contact the Oracle Support Services Representative.

ORA-05195: bad or missing kernel connector

Cause: The loaded kernel could not verify its user-side connector module. This can occur if you use an incorrect kernel version.

Action: If you cannot identify the cause of the problem, then contact the Oracle Support Services Representative.

ORA-05198: associated internal OSD error code %d

Cause: This message precedes ORA-05199, if there is more information available. The first 4 hexadecimal digits can often identify the module, and the last 4 hexadecimal digits are usually a condensed version of an associated system macro code. This code can be helpful in diagnosing the problem.

Action: If you cannot identify the cause of the problem, then contact the Oracle Support Services Representative.

ORA-05199: ORACLE ABNORMAL EXIT

Cause: A fatal error occurred, which prevents continuation of execution. In many cases, a preceding message explains the error. The system causes the program execution to stop (`TERM ABNORMAL` with `DUMP` is displayed).

Action: If you cannot identify the cause of the problem, then contact the Oracle Support Services Representative.

B

Oracle Environment Variables

This appendix describes variables that can be specified in the ORAENV file or the POSIX shell. Oracle parameters, such as ORACLE_SID and NLS_LANG, may be specified in the ORAENV file or POSIX shell.If you use a ORAENV file, then you must follow the ORAENV rules for specifying environment variables as described in the following sections. In the POSIX shell, you must follow the UNIX rules to set and export the environment variables.

The following table describes the variables that are categorized into three classes.

Class	Description
DBA	These variables are for database administration purposes. Most DBA variables are evaluated only during database startup.
USER	These variables can be specified by ordinary users as well as by the DBA. When these variables are specified in a particular user's ORAENV file, they modify that user's environment only.
NET	These variables apply to Oracle Net Services components. These variables should be included in the ORAENV file of the DBA.

The class (or classes) to which a variable belongs is noted in the variable descriptions in this appendix.

Any DBA or NET variables specified in an ordinary user's ORAENV file are ignored.

B.1 ORAENV Rules

You should consider the following general rules when writing ORAENV files:

- All lines which begin with a slash or asterisk (/ or *) are ignored.
- All variable names must be written in uppercase.
- Spaces must not be included immediately before and after the equals sign (=).
- Do not enclose values in quotation marks unless you want the quotation marks to be part of the value.
- Since the variable list is conceptually open ended, errors in variable names are not recognized. This means that the value of any variable whose name is mis-typed is not modified.
- There is only limited checking of variable assignments. An incorrect value may generate an error message, but may also be interpreted as a null value.

- When variable assignments refer to other variables, BS2000 command file substitution syntax applies. Substitution takes place when a variable is actually used, not when it is read from the ORAENV file.

  For example:

  ```
  ORAUID=$ORAC1120
  SQLPATH=&ORAUID..RDBMS.ADMIN
  ```

  assigns the value $ORAC1120.RDBMS.ADMIN to the variable SQLPATH. If ORAUID is changed, then SQLPATH automatically reflects the new value.
- The sequence of items in the ORAENV file is not generally significant. If an item occurs more than once, then the last occurrence is used.
- If no value is given for a variable, then the default value is used, if it exists.

B.2 Built-in Variables

The following variables are always defined, and may be referenced in other variable assignments:

B.2.1 LOGNAME

The LOGNAME variable always contains the current BS2000 user ID. You cannot alter the value of this variable by assigning a different value to it in the ORAENV file.

B.2.2 ORAUID

This variable specifies the BS2000 user ID where the Oracle Database programs, installation and demonstration files are installed. The initial value is derived from the ORALOAD link name (the user ID part of the ORALOAD library name). This value is usually correct, but if necessary, you can override it by assigning a different value to it in the ORAENV file.

Format: ORAUID=*$userid* or ORAUID=/BS2/*$userid*

B.2.3 PGM

The PGM variable always contains the last part of the current START_PROGRAM program name. You cannot alter the value of this variable by assigning a different value to it in the ORAENV file.

B.2.4 TERM

The TERM variable contains the terminal type, and defaults to 'SNI9750'. This default value is usually correct, but if necessary, you can override it by assigning a different value to it in the ORAENV file.

B.2.5 TSN

The TSN variable contains the task sequence number of the current task. You cannot alter the value of this variable by assigning a different value to it in the ORAENV file.

B.3 General Variables

The following variables are for general, day-to-day use by Oracle DBAs and users.

B.3.1 CLN_BASE

This variable specifies the address of the shared code pool of customer written database applications for CORE, NLS, and NET.

Format:

CLN_BASE=*address*

Classification:

USER

Default:

CLN_BASE=37M

B.3.2 CLN_MPID

This variable specifies the identification of the shared code pool of customer written database applications for CORE, NLS, and NET.

Format:

CLN_MPID=*sid*

Classification:

USER

Default:

CLN_MPID=*&ORASID*

B.3.3 DEFAULT_CONNECTION

This variable provides a default host string for connect requests where no host string is specified. If you always connect to the same database, then it may be convenient to specify this. This value should contain everything you would otherwise specify after the "@" character.

Format: DEFAULT_CONNECTION=*host-string*

Classification: USER

Example:

```
DEFAULT_CONNECTION=TNS:
(DESCRIPTION=
(ADDRESS=
(PROTOCOL=TCP)
(HOST=MADRID)
(PORT=1521))
(CONNECT_DATA=
(SERVICE_NAME=PROD)))
```

B.3.4 EXP_CLIB_FILE_IO

This variable should be set to FALSE when you use the Export utility to overcome a problem with the C library functions when an export file is written to tape.

Format: EXP_CLIB_FILE_IO=FALSE

Classification: USER

Default: EXP_CLIB_FILE_IO=TRUE

B.3.5 IMP_CLIB_FILE_IO

This variable should be set to FALSE when you use the Import utility to overcome a problem with the C library functions when an import file is read from tape.

Format: IMP_CLIB_FILE_IO=FALSE

Classification: USER

Default: IMP_CLIB_FILE_IO=TRUE

B.3.6 NLS_LANG

This variable specifies the default language and character set. For example:

```
NLS_LANG=GERMAN_GERMANY.D8BS2000
```

Format: NLS_LANG=*language_territory.character-set*

Classification: USER, DBA

Default: NLS_LANG=AMERICAN_AMERICA.WE8BS2000

B.3.7 OPS_JID

This variable is used for concatenation with the OS_AUTHENT_PREFIX, refer to initialization parameter. The default value concatenates the value of the parameter OS_AUTHENT_PREFIX with the BS2000 user ID. Using OPS_JID, you can specify that the BS2000 jobname, /.jobname LOGON..., is used instead. This is useful when many users are sharing one BS2000 user ID.

Format: OPS_JID=*userid/jobname*

Classification: DBA

Default: *userid*

B.3.8 ORASID

This variable defines the database that is used if no database identification is given at connect time.This variable is a synonym of the ORACLE_SID variable.

Format: ORASID=*sid* (*sid* is a characterstring where 1 <= length <= 4)

Classification: USER, DBA

Note: Oracle recommends that you use the ORACLE_SID variable.

B.3.9 PRINTPAR

This variable specifies optional variables for the /PRINT command issued for SPOOL OUT spool files. Using this variable, the user can modify the spooled job, and, for example, route the job to a remote printer, add print options for laser printers, and so on. The BS2000 /PRINT command for spool files is issued as follows:

```
/PRINT temp.spoolfile,&PRINTPAR
```

Format: PRINTPAR=*print-options*

Classification: USER

B.3.10 SQLPATH

This variable specifies a path where SQL*Plus looks for command files. Elements of the path are separated by semicolons (;). For example:

```
SQLPATH=PRIVATE;$ORAC1120
```

This assignment causes SQL*Plus to look for filename.SQL, then for PRIVATE.*filename*.SQL, and finally for $ORAC1120.filename.SQL.

Format: SQLPATH=*search-path*

Classification: USER, DBA

B.3.11 SSSIDPWF

This variable specifies the password file for remote instance start. For further information, refer to Chapter 5, "Administering Oracle Database".

Format: SSSIDPWF=*password-file*

Classification: DBA

B.4 DBA Startup Variables

The following variables are used during database and network startup. They supplement (and in some cases provide defaults for) variables contained in the initialization file.

Oracle recommends that database startup and shutdown, background jobs, and network jobs should all refer to the same ORAENV file to ensure that the variables are consistent.

Note that the default values listed in the following section are built-in defaults, most of them are over-ridden by settings in the shipped DEMO.P.ORAENV.

B.4.1 Address and Size Specification

Several of the variables described in this section define memory addresses and sizes. The notation used to specify these items is as follows:

- A number with no modifiers is interpreted as a decimal number
- A number followed by K or M is interpreted as a decimal number multiplied by 1024 or 1048576 (1024*1024) respectively
- A number enclosed in single quotation marks and preceded by the letter X is interpreted as a hexadecimal number

For example, the following all set the KNL_BASE variable to 8M:

```
KNL_BASE=8M
KNL_BASE=8388608
KNL_BASE=X'800000'
```

B.4.2 BGJPAR

This variable specifies the parameters for the ENTER-PROCEDURE command used when starting background jobs. The ENTER-PROCEDURE command is used to submit jobs as follows:

```
.jobname ENTER-PROCEDURE jobfile, &BGJPAR
```

Format: BGJPAR=*parameters*

Classification: DBA

> **Note:** The BGJPAR variable is set up by the installation procedure.

B.4.3 BGJPRC_UID / BGJPRC_SID

These variables specify the user ID and orasid of the file for the background enter jobs. If the use of a special enter job file is desired, then the parameters must be set to the desired userid and orasid.

Format:

```
BGJPRC_UID=$userid
BGJPRC_SID=sid
```

Classification: DBA, NET

Default:

```
BGJPRC_UID=&ORAUID
BGJPRC_SID=DEMO
```

B.4.4 BGJ_LOG_JOBSTART

This variable specifies whether the operating system message that a new job was accepted should be logged on SYSOUT or not.

Format: BGJ_LOG_JOBSTART=Y/N

Classification: DBA,USER, NET

Default: BGJ_LOG_JOBSTART=N

B.4.5 sid_BGJPAR

This variable specifies the parameters, which are used by the ENTER-PROCEDURE command to start a server process for the instance specified by *SID*.

Format: *sid*_BGJPAR=*parameters*

Syntax: *sid* is a string of at the most 4 alphanumeric characters

parameters is the parameters for the ENTER-PROCEDURE command as described in the BS2000/OSD commands

Classification: DBA,USER, NET

B.4.6 sid_USER

This variable specifies the USER-ID where the instance assigned by *SID* resides.

Format: *sid*_USER=*userid*

Syntax: *sid* is a string of at the most 4 alphanumeric characters

userid is a string of at most 8 alphanumeric characters which follows the rules of a BS2000/OSD USER-ID

Classification: DBA,USER, NET

B.4.7 user_ACCOUNT/ user_PASSWORD

*user*_ACCOUNT or *user*_PASSWORD define credentials of a USER-ID, which are used by the ENTER-PROCEDURE command to start a process.

Format: *user*_ACCOUNT=*account*

*user*_PASSWORD=*password*

Syntax: *user* is a string of at the most 8 alphanumeric characters, which follows the rules of a BS2000/OSD USER-ID and must match a USER-ID defined by the parameter *sid*_USER.

account is a string of at the most 8 alphanumeric characters, which follows the rules for a BS2000/OSD account number.

password is a string of at the most 8 alphanumeric characters, which follows the rules for a BS2000/OSD password.

Classification: DBA, NET

B.4.8 COM_MPID

This parameter specifies the identification of the shared code pool of the Oracle instance for CORE, NLS, and NET.

Format: COM_MPID=*sid*

Classification: DBA

Default: COM_MPID=*&ORASID*

B.4.9 COM_BASE

This parameter specifies the address of the shared code pool of the Oracle instance for CORE and NLS.

Format: COM_BASE=*address*

Classification: DBA

Default: COM_BASE=*37M*

B.4.10 JOBID

This variable is used internally in identifying the background tasks and generating task-specific names. You see it in some places, but you should never specify it yourself.

Classification: DBA

B.4.11 KNL_BASE

This variable gives the base address where the shared memory pool is mapped in memory. This must be an integral number of megabytes.

Format: KNL_BASE=*address*

Classification: DBA

Default: KNL_BASE=72M

B.4.12 ORACLE_HOME

The Oracle home directory is the directory in the POSIX file system which contains the installation of the software for a particular Oracle product.

Format: ORACLE_HOME=*/path-name*

Classification: DBA, USER

B.4.13 PGA_BASE

This variable specifies the base address of the fixed part of the PGA. The PGA is task-specific, but must be located at a fixed memory address so that the kernel can access it. The base address must lie on a 64KB boundary.

Format: PGA_BASE=*address*

Classification: DBA

Default: PGA_BASE=189M

Note: The value of PGA_BASE is taken from the kernel if the shared kernel is already loaded.

B.4.14 PGA_SIZE

This variable specifies the size of the fixed part of the PGA. This variable should not be changed from its default value.

Format: PGA_SIZE=*size*

Classification: DBA

Default: PGA_SIZE=64K

Note: The value of PGA_SIZE is taken from the kernel if the shared kernel is already loaded.

B.4.15 SF_PBLKSIZE

This variable specifies the physical blocksize of redo log files.

Format: SF_PBLKSIZE=2K|4K

Classification: DBA

Default: 2K

Note: This variable cannot be changed after database creation. Once you specify a value different from the default, you must specify it in all future calls.

B.4.16 SGA_BASE

This variable gives the address where the SGA is mapped into memory, and must represent a megabyte-boundary.

Format: SGA_BASE=*address*

Classification: DBA

Default: SGA_BASE=190M

Note: The value of SGA_BASE is read from the kernel if the shared kernel is already loaded. There is no corresponding SGA_SIZE variable; the size of the SGA memory pool is calculated when the database is started.

B.5 Oracle Net Services Variables

The following are the Oracle Net Services variables:

B.5.1 BREAK_HANDLING

This variable deactivates the signal routine for user interrupts, which sends a break over the network. An interrupt can be released by pressing the [K2] key.

Format:

BREAK_HANDLING=ON|OFF

Classification:

DBA, USER, NET

Default:

BREAK_HANDLING=ON

B.5.2 TNS_ADMIN

This variable specifies the user ID of the Oracle Net Services configuration files, for example, LISTENER.ORA, TNSNAMES.ORA and SQLNET.ORA. If TNS_ADMIN is not defined, then the configuration files are searched under the local user ID with the prefix NETWORK.ADMIN.

Format: TNS_ADMIN=$userid

Classification: DBA, USER, NET

B.5.3 TNS_BEQ_TIMEOUT

This variable specifies the time after which a connection between a parent and a child process is closed if there is no communication between them.

Format: TNS_BEQ_TIMEOUT=*lifetime* (in seconds)

Classification: NET

Default: TNS_BEQ_TIMEOUT=180

B.5.4 TNS_UPDATE_IPNODE

This variable forces the Oracle Net software to change always the server's IP-Node name to an IP-Node address.

Format: `TNS_UPDATE_IPNODE=TRUE/FALSE`

Classification: `NET`

Default: `TNS_UPDATE_IPNODE=FALSE`

C

Initialization Parameters and the Parameter File

Every time SQL*Plus starts an Oracle Database instance, it uses a set of parameters which specify the characteristics of the instance's operation. These parameters are kept in a file, typically named *sid*.DBS.INIT.ORA.

This appendix lists unsupported parameters, and lists other parameters that you may need to change to customize the Oracle Database for the system.

Refer to the *Oracle Database Reference* manual for general descriptions of the parameters listed in this Appendix.

C.1 Example Parameter File

The $ORAC1120.DEMO.DBS.INIT.ORA parameter file is created upon initial installation and can be edited as a text file.

C.2 Unsupported Parameters

The following initialization file parameters, described in the generic documentation are not supported by Oracle Database 11*g* for BS2000/OSD.

- MAX_DUMP_FILE_SIZE
- OS_ROLES
- AUDIT_SYSLOG_LEVEL
- MEMORY_MAX_TARGET
- MEMORY_TARGET

Specifying these parameters in the initialization file results in an Oracle Database error during startup. The workaround is to remove such lines from the file.

C.3 Additional Notes on Initialization Parameters

This section contains additional information about initialization parameters.

C.3.1 BACKGROUND_DUMP_DEST

This parameter specifies the path name (directory or prefix) where debugging trace files for the background processes (LGWR, DBWn, and so on.) are written during Oracle operations. Furthermore, it specifies the path name for the alert file. The default

value for this parameter is the current BS2000 user ID of the Oracle background processes. You can specify a prefix for the trace and alert files in the following format:

```
BACKGROUND_DUMP_DEST=BDD
```

You can also specify a POSIX directory for this parameter, if you have enabled the POSIX subsystem.

> **Note:** This parameter is ignored by the new diagnosability infrastructure introduced in Oracle Database 11*g*, which places trace and core files in a location controlled by the `DIAGNOSTIC_DEST` initialization parameter

C.3.2 USER_DUMP_DEST

This parameter specifies the path name (directory or prefix) where the server writes debugging trace files on behalf of a user process. The default value for this parameter is the current BS2000 user ID of the Oracle Database processes.

You can specify a prefix for the trace files as follows:

```
USER_DUMP_DEST=UDD
```

You can also specify a POSIX directory for this parameter, if you have enabled the POSIX subsystem.

> **Note:** This parameter is ignored by the new diagnosability infrastructure introduced in Oracle Database 11*g*, which places trace and core files in a location controlled by the `DIAGNOSTIC_DEST` initialization parameter

C.3.3 AUDIT_FILE_DEST

This parameter specifies the path name (directory or prefix) into which the audit trail is written when the `AUDIT_TRAIL` initialization parameter is set to `OS`. Usually this value is used as a prefix for BS2000 file names. You can also specify a POSIX directory for this parameter, if you have enabled the POSIX subsystem. The default value for this parameter is *`SID`*. ADUMP. The name of the audit files is `tsn-seqno.AUD`, where *`tsn`* is the task sequence number of the current task and *`seqno`* is a sequence number. Bear in mind that regardless of whether database auditing is enabled, Oracle/BS2000 always records some database-related actions into the operating system audit file: instance startup, shutdown and connections with administrator privileges.

C.3.4 DB_BLOCK_SIZE

This parameter can have one of the following values:

- 2K, 4K, 6K, 8K, 16K, 32K, if you use BS2000 2K pubset format.
- 4K, 8K, 16K, 32K, if you use BS2000 4K pubset format.

C.3.5 DB_FILE_MULTIBLOCK_READ_COUNT

The default value of this parameter is 64K/`DB_BLOCK_SIZE`, which is also the maximum value. Setting this parameter beyond this limit has no effect.

C.3.6 DB_FILES

If you plan to create a large database, then you must set this value to the maximum of 2044 before creating the database.

C.3.7 LOCK_SGA

This parameter is ignored on Oracle Database 11*g* for BS2000/OSD. Buffers in the SGA are page fixed only during I/O operations. Otherwise, the SGA on BS2000 is pageable.

C.3.8 SGA_MAX_SIZE

This parameter should not be specified on Oracle Database 11*g* for BS2000/OSD. Because the SGA is not permanently pagefixed as it is on some other systems, there is little benefit in reserving SGA expansion space with the SGA_MAX_SIZE parameter. It defaults to the actual SGA size.

C.3.9 LOG_ARCHIVE_BUFFER_SIZE

The value of this parameter should always be set so that when multiplied by the value of `SF_PBLKSIZE` the result equals 32K.

C.3.10 LOG_ARCHIVE_DEST

This parameter can indicate a pubset, such as `LOG_ARCHIVE_DEST=:PUB1:` to store all archived redo logs on special media.

D

Troubleshooting

This section describes problems that you may encounter when using the Oracle Database 11*g* Release 2 on BS2000, and provides you with information about how to diagnose and overcome such problems.

To solve a problem, identify the type of the problem and locate the relevant information in this appendix. Examine each of the listed points to find the cause of the problem. Carry out the suggested solution, and try again. The event log file described in this document may help you to diagnose the problem.

Refer to the Appendix A, "Oracle Error Messages for BS2000/OSD" in this guide and *Oracle Database Error Messages* manual for information about specific messages.

D.1 Problems Installing Oracle Database 11*g*

The following sections describe some probable problems faced while installing Oracle Database 11*g* Release 2.

D.1.1 Problems Creating a Database

You should always use `INSTALL.P.SUPER` or `INSTALL.P.DBA` to create a new database, because this is the easiest way to get a correct instance. If you encounter problems during this process, then study the diagnostic output, correct, and run the respective part manually, or remove the partially-created database, and re-run the whole process. All files belonging to a specific database are prefixed with the system identifier (`ORASID`) for that database, except for log files which have an extra prefix.

Also, check the following:

- Does the BS2000 user ID have an adequate `PUBLIC-SPACE-LIMIT` for the corresponding pubset?
- Is enough disk space available on the pubset or disks used to create the databases?
- Is disk space fragmentation too high?

D.2 Problems Starting a Database

The following section lists some probable issues that you might face when you start a database.

D.2.1 Problems Starting a Database

This section lists information related to problems encountered when starting a database.

- Did you get an ORA-05032 error with no extra information?

  When you attempt to start a database and the startup fails, you sometimes get an ORA-05032 message and not much other information. This indicates that a problem occurred in a very early stage of the startup, when Oracle Database 11*g* error stack and backtracking mechanism was not yet active. If this is the case, then you should check the following:

  – Did you call the `ORAENV` procedure prior to calling SQL*Plus?

  – Did you specify a correct and unique `ORASID` value in the `ORAENV` file?

  – Are there potential address range conflicts?

    The address ranges assigned to the kernel memory pool, the SGA, and the PGA, in each task, could be partially occupied by shared subsystems also used in the instance. Contact the System Administrator to find out how the subsystems are arranged. Then change the corresponding `xxx_BASE` environment variables in the `ORAENV` file to relocate the Oracle Database 11*g* areas to suitable address ranges.

  – Is the user address space large enough?

    A small address space limit may not leave enough room for Oracle Database 11*g* requirements.

  – Has a previous startup attempt failed, leaving invalid background, database, or user tasks?

    If the Oracle Database 11*g* has not been shut down properly, then old background or database tasks may hang and still be connected to the SGA of the old instance. This inhibits the creation of a new SGA. You may get a message indicating `shutdown in progress`.

    Cancel the remaining background, server, and user tasks. Exit SQL*Plus (this is required to release shared memory pools of the old instance) and retry.

D.2.2 Problems with Tasks

If you get a time out message when starting the background tasks, then check the following points:

- Are the background tasks blocked in the BS2000 job queue? This may occur due to system overload or insufficient task priority.

  The background tasks should always be started with the `IMMEDIATE` option and preferably in a reserved `Jobclass`. Check the `ORAENV BGJPAR` environment variable and the user attributes of the BS2000 user ID. Cancel any background tasks that have already started.

- If no background task can be found using the `/STATUS` command, then the jobs have probably aborted. Check the job outputs.

D.3 Problems Accessing the Database

Refer to this section if you are facing issues accessing the database.

D.3.1 Problems with Database and Log files

If you have problems opening, closing, reading, or writing a database or log file, then check the following points:

- Does the file exist?
- Is the file accessible to the program which is trying to open it?
- Is there a hardware problem?
- Did you specify the correct block size?

If you specified the ORAENV environment variable, SF_PBLKSIZE, at database creation, then you must continue to use the same specification whenever you run an ALTER DATABASE statement.

D.3.2 Oracle Database 11*g* Trace Files

Whenever Oracle Database 11*g* encounters an exception, it writes a trace (or dump) file. You may need to send the file to the Oracle Support Services Representative if any unusual problem occurs.

These files are created at database startup with a standard header and are modified for the last time at database shutdown. If no problems have occurred, then you may want to remove these files after a successful shutdown.

D.3.3 Oracle Database-Level Error Information

When you get an Oracle Database message, the ORA-xxxx message may sometimes be followed by a message like the following:

```
SOSD error 8xxxyyyy from mmmmmmmm : text
```

This indicates that the error originated in operating system code or low-level Oracle Database code interfacing with the operating system. The SOSD error code provides important diagnostic information, and when contacting the Oracle Support Services Representative you should always supply this code, if present, in addition to the Oracle Database error number.

The error code is displayed in hexadecimal, and is structured as follows:

- *xxx* identifies the function reporting the error. This information is useful to the Oracle Support Services Representative.
- *yyyy* details the error. It is either an internal code of the function, or a compacted return code of a BS2000 system macro (see subsequent section).
- *mmmmmmmm* is the name of the Oracle Database internal function. Text, if present, explains the error code. Often it says "RC FROM zzzzz MACRO".

A BS2000 system macro return code is condensed into the 2-byte value yyyy as follows:

- For system macros that return a code bb0000aa, *yyyy* is bbaa
- For I/O calls, *yyyy* is the DMS error code

In all other cases, *yyyy* contains the right halfword of the return code of the BS2000 macro.

E

File Types and Names Used by Oracle

The following is a list of file types and names used by Oracle.

sid.DBS.xxx.DBF

Database files such as *sid*.DBS.CONTROL.DBF, *sid*.DBS.DATABASE1.DBF, or *sid*.DBS.LOG1.DBF contain the entire Oracle database including data dictionary, user tables, log files, and so on.

sid.DBS.INIT.ORA

Parameter file used when instance is started.

sid.P.ORAENV

Environment file containing user environment. Always to be run before start of an instance or an application: /DO *sid*.P.ORAENV.

S.E.tsn.YYYY-MM-DD.hh.mm.ss

Temporary ENTER-PROCEDURE file for starting background processes (using $ORAC1120.DEMO.P.ENTER).

S.OUT.tsn.YYYY-MM-DD.hhmmss

Temporary files of background processes containing run-time information. After a task is successfully completed, these files are removed. These files can help diagnosis.

L.sid.xxxx.SYSOUT.tsn

Run-time listing documenting the start of a background process. When process finishes orderly listings are removed, except the SDF-P variable BJGOUT is set to KEEP in *sid*.P.ORAENV for diagnosis purposes. *xxxx* corresponds with the identifier of Oracle background jobs.

Oracle Net Services LOG files

All errors encountered in Oracle network products are appended to a log file. The default directory path on BS2000/OSD is network.log. For more information about Oracle Net Services log files, refer to *Oracle Database Net Services Administrator's Guide*.

T.sid.INSTALL.E.SUPER

To be removed after successful installation.

T.sid.INSTALL.E.SUPER.72

To be removed after successful installation.

L.sid.xxx.LOG

To be removed after successful installation, for example, `L.sid.CATALOG.LOG`.

ORAXALOG.tsn-NULL-YYMMDD.TRC

This file is used to trace errors when openUTM is used with Oracle. This file is created only when an error occurs. `NULL` can be replaced by a `db_name` when specified in the open string.

Index

A

Address Space, 2-5
administration utility
  SQL*Plus, 5-1
Architecture
  Oracle Server, 2-1
Archiving, 1-4
AUDIT_FILE_DEST, C-2

B

BACKGROUND_DUMP_DEST, C-1
BCAM, 9-10
Bequeath adapter, 9-3
BGJ_LOG_JOBSTART, B-6
BGJPAR, 7-1, B-6
Bigfile Tablespaces, 2-5

C

CE8BS2000, 12-2
CIRCUITS, 9-4
CL8BS2000, 12-2
Client Common Pool, 2-6
CLN_BASE, B-3
CLN_MPID, B-3
COM_BASE, B-7
Creating a database, 4-1

D

Data dictionary views
  installing, 4-7
Database
  creating, 4-1
  files, 2-2
  recovering, 6-2
  system identifier, 2-7
Database character sets
  Java, 12-2
DB_BLOCK_SIZE, C-2
DB_FILE_MULTIBLOCK_READ_COUNT, C-2
DB_FILES, C-3
DEFAULT_CONNECTION, B-3
DISPATCHERS, 9-4

E

EE8BS2000, 12-2
Environment Definition File, 2-6
Environment variables
  BGJPAR, 7-1, B-6
  CLN_BASE, B-3
  CLN_MPID, B-3
  DEFAULT_CONNECTION, B-3
  EXP_CLIB_FILE_IO, B-3
  IMP_CLIB_FILE_IO, B-4
  JOBID, B-7
  KNL_BASE, B-7
  LOGNAME, B-2
  NLS_LANG, B-4
  OPS_JID, B-4
  ORAUID, B-2
  PGA_BASE, B-8
  PGA_SIZE, B-8
  PGM, B-2
  PRINTPAR, B-4
  SF_PBLKSIZE, B-8
  SGA_BASE, B-9
  SQLPATH, B-5
  SSSIDPWF, B-5
  TERM, B-2
  TNS_ADMIN, B-9
  TSN, B-2
EXP_CLIB_FILE_IO, B-3
Export, 1-4
External procedure calls, 11-1

F

File Names, E-1
File Types, E-1
Files
  database, 2-2
  INIT.ORA, C-1
  ORAENV, B-1
  trace, 8-8
files
  Java installation, 12-1

G

Globalization Support, 1-4
GLOGIN.SQL, 5-3

H

HOST, 5-2

I

IMP_CLIB_FILE_IO, B-4
Import, 1-4
Initialization parameters
  CIRCUITS, 9-4
  DB_FILE_MULTIBLOCK_READ_COUNT, C-2
  DB_FILES, C-3
  DISPATCHERS, 9-4
  INIT.ORA file, C-1
  LOCAL_LISTENER, 9-4
  MAX_DISPATCHERS, 9-4
  MAX_SHARED_SERVERS, 9-4
  SHARED_SERVER_SESSIONS, 9-4
  SHARED_SERVERS, 9-4
INIT.ORA, 2-3, C-1
  modifying, 4-6
Installation
  Java Enabled Database, 12-1
Intelligent Agent, 14-1
Invoking
  SQL*Plus, 5-1

J

Java, 12-1
  database character sets, 12-2
JOBID, B-7

K

KNL_BASE, B-7
Known problems, restrictions and workarounds, 1-2

L

LDAP.ORA, 9-7
Loading External Procedures, 11-1
LOCAL_LISTENER, 9-4
LOCK_SGA, C-3
Log Files, 2-2
LOG_ARCHIVE_BUFFER_SIZE, C-3
LOG_ARCHIVE_DES, C-3
LOGIN.SQL, 5-3
LOGNAME, B-2
L.sid.xxx.LOG, E-2
L.sid.xxxx.SYSOUT.tsn, E-1

M

Making PL/SQL available, 4-7
MAX_DISPATCHERS, 9-4
MAX_SHARED_SERVERS, 9-4
Messages, A-1
Modifying INIT.ORA, 4-6

N

NLS_LANG, B-4

O

*open*Net Server
  netstat, 14-1
  tool, 14-1
OPS_JID, B-4
Oracle
  file types, E-1
Oracle Database
  administering, 5-1
  architecture, 2-1
  tuning, 7-1
Oracle Database Environment Definition File, 2-6
Oracle Net Services LOG files, E-1
Oracle Text, 10-1
ORACLE_HOME, B-8
ORAENV, 2-3, 2-6, 2-7, 4-6, B-1
ORAENV variables, 2-7
ORALOAD.LIB, 2-3
ORAUID, B-2
ORAXALOG.tsn-NULL-YYMMDD.TRC, E-2

P

parameter files, 5-3
Parameters
  initialization, C-1
PGA, B-8
PGA_BASE, B-8
PGA_SIZE, B-8
PGM, B-2
PL/SQL
  making available, 4-7
Precompilers
  interfacing to the Oracle XA Library, 8-7
Preinstallation Issues, 14-1
Printers, B-4
PRINTPAR, B-4
Problems, 1-2
Protocol Adapters
  TCP/IP, 9-3

R

Recovering Databases, 6-2
Remote Startup
  database instance
    remote startup, 5-5
Restrictions, 1-2

S

Server Parameter File, 5-4

S.E.tsn.YYYY-MM-DD.hh.mm.ss, E-1
SF_PBLKSIZE, B-8
SGA_BASE, B-9
SGA_MAX_SIZE, C-3
shared code pool, B-7
SHARED_SERVER_SESSIONS, 9-4
SHARED_SERVERS, 9-4
sid_BGJPAR, B-6
SID_LIST_LISTENER, 5-5
sid_USER, B-6
sid.DBS.INIT.ORA, E-1
sid.DBS.xxx.DBF, E-1
sid.P.ORAENV, E-1
S.OUT.tsn.YYYY-MM-DD.hhmmss, E-1
SPFILE, 2-3, 5-4
SQ130, 15-1
SQL*Net TCP/IP, 1-5
SQL*Plus, 1-4, 5-1
  and INIT.ORA, 5-4
  calling from procedures, 5-2
  database administration utility, 5-1
  invoking, 5-1
  issuing BS2000 commands from, 5-2
SQLPATH, B-5
SSSIDPWF, B-5
Starting
  Oracle Text, 10-1

T

TCP/IP, 1-5
TCP/IP adapter, 9-3
TERM, B-2
timeout handling, D-2
TNS_ADMIN, B-9
TNS_UPDATE_IPNODE, B-10
Trace Files, 8-8
Troubleshooting, 8-8
T.sid.INSTALL.E.SUPER, E-1
T.sid.INSTALL.E.SUPER.72, E-1
TSN, B-2
Tuning
  Oracle Database, 7-1

U

User ID
  DBA, 2-10
  Oracle users, 2-11
user ID
  installation, 2-9
user_ACCOUNT, B-7
USER_DUMP_DEST, C-2
user_PASSWORD, B-7
Utilities
  Import, 1-4
UTM, 8-1

V

variables
  ORAENV, 2-7

W

WE8BS2000, 12-2
WE8BS2000E, 12-2
WE8BS2000L5, 12-2
Workarounds, 1-2

X

XAO.LIB, 8-1
XDK, 13-1
XML, 13-1
XML parser, 13-1
XSLT Processor, 13-1